Uncertainty In Artificial Intelligence

Proceedings of the Twenty-Ninth Conference (2013)

Edited by

Ann Nicholson

Padhraic Smyth

Uncertainty in Artificial Intelligence

Proceedings of the Twenty-Ninth Conference (2013)

July 12-14, 2013, Bellevue, Washington, United States

Edited by
Ann Nicholson, Monash University, Australia
Padhraic Smyth, University of California, Irvine, United States

Sponsored by
Microsoft Research, Artificial Intelligence Journal, Amazon.com, Google Inc., Charles River Analytics, Toyota InfoTechnology Center, U.S.A., Inc., IBM Research

AUAI Press Corvallis, Oregon

Published by AUAI Press for
Association for Uncertainty in Artificial Intelligence
http://auai.org

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

ISBN 978-0-9749039-9-6

Contents

Preface

This volume contains all papers that were accepted for the 29th Conference on Uncertainty in Artificial Intelligence (UAI), held in Bellevue, Washington, USA, from July 12 to 14th 2013. 233 papers were submitted to the conference and each was peer-reviewed by 3 or more reviewers. From the 233 papers, a total of 73 papers were accepted, 26 for oral presentation and 47 for poster presentation, for an acceptance rate of 31%. We are very grateful to the senior program committee and program committee members for their diligence in generating over 700 reviews for the 233 submitted papers.

In addition to the presentation of technical papers, the following invited speakers were also scheduled to give keynote talks at UAI 2013: Tom Mitchell (Carnegie Mellon University), Ralf Herbrich (Amazon), and Josh Tenenbaum (MIT).

A day of UAI 2013 tutorials was organized by tutorials chair David Sontag, on topics such as Computational Advertising and Causality (Leon Bottou), Large-Scale Distributed Learning with GraphLab (Carlos Guestrin), Statistical Methods in Genomics (Lior Pachter), and Polynomial Methods in Learning and Statistics (Ankur Moitra). UAI 2013 also featured a day of workshops, organized by workshops chair John Mark Agosta, on topics such as Approaches to Causal Structure Learning, Big Data meet Complex Models, New Challenges in E-Commerice Recommendations, and Models for Spatial, Temporal, and Network Data.

Ann Nicholson and Padhraic Smyth (Program Co-Chairs)
Nando De Freitas and Max Chickering (General Co-Chairs)

Organizing Committee

Program Chairs

Ann Nicholson, Monash University, Australia
Padhraic Smyth, University of California, Irvine, USA

General Chairs

Nando De Freitas, University of British Columbia, Canada, and Oxford University, UK
Max Chickering, Microsoft Research, Redmond, USA

Local Arrangements Chair

Marina Meila, University of Washington, USA

Proceedings Chair

Alice Zheng, Microsoft Research, Redmond, USA

Tutorials Chair

David Sontag, New York University, USA

Workshops Chair

John Mark Agosta, Toyota Information Technology Center, Mountain View, 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 2013 were responsible for generating reviews and recommendations for the 233 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

Adnan Darwiche	UCLA, USA
Denver Dash	Intel Labs, Pittsburgh, USA
Rina Dechter	University of California, Irvine, USA
Francisco J. Diez	UNED, Spain
Norman Fenton	Queen Mary University, London, UK
Alan Fern	Oregon State University, USA
Emily Fox	University of Washington, USA
Lise Getoor	University of Maryland, USA
Alexander Ihler	University of California, Irvine, USA
Tommi Jaakkola	MIT, USA
Dominik Janzing	Max Planck Institute for Biological Cybernetics, Germany
Kristian Kersting	Fraunhofer IAIS, University of Bonn, Germany
Helge Langseth	The Norwegian University of Science and Technology, Norway
Kathryn Laskey	George Mason University, USA
Guy Lebanon	Georgia Tech/Amazon, USA
Tze Yun Leong	National University of Singapore
Kevin Leyton-Brown	University of British Columbia, Canada
Chris Meek	Microsoft Research, USA
Brian Milch	Google, USA
Remi Munos	INRIA Lille, France
Petri Myllymaki	Helsinki Institute for Information Technology, Finland
Thomas D. Nielsen	Aalborg University, Denmark
Nuria Oliver	Telefonica, Spain
Ronald Parr	Duke University, USA
Avi Pfeffer	Charles River Analytics, USA
David Poole	University of British Columbia, Canada
Greg Provan	University College Cork, Ireland
Thomas Richardson	University of Washington, USA
Dale Schuurmans	University of Alberta, Canada
Christian Shelton	University of California, Riverside, USA
Prakash Shenoy	University of Kansas, USA
Ricardo Silva	University College London, UK
Alex Smola	Carnegie Mellon University, USA

Peter Spirtes	Carnegie Mellon University, USA
Umberto Straccia	ISTI-CNR, Italy
Jin Tian	Iowa State University, USA
Yair Weiss	Hebrew University, Israel
Michael Wellman	University of Michigan, USA
Dit-Yan Yeung	The Hong Kong University of Science and Technology, Hong Kong
Nevin Zhang	The Hong Kong Unversity of Science and Technology, Hong Kong

Program Committee

Ryan P. Adams	Harvard University, USA
Arvind Agarwal	University of Maryland, USA
Babak Ahmadi	Fraunhofer Institute, Germany
Amr Ahmed	Google, USA
Ayesha Ali	University of Guelph, Canada
Russell Almond	Florida State University, USA
Animashree Anandkumar	University of California, Irvine, USA
Dragomir Anguelov	Google, USA
Alessandro Antonucci	IDSIA, Switzerland
Cedric Archambeau	Xerox Research Centre Europe, France
Nimar Arora	Bayesian Logic, Inc., USA
Arthur Asuncion	Google, USA
Elias Bareinboim	UCLA, USA
Dhruv Batra	Virgina Tech, USA
Salem Benferhat	CRIL, University of Artois, France
Carlo Berzuini	University of Manchester, UK
Alina Beygelzimer	IBM Research, USA
Bozhena Bidyuk	Google, USA
Charles Blundell	Gatsby Unit, UCL, UK
Fernando Bobillo	Universidad de Zaragoza, Spain
Guillaume Bouchard	Xerox Research Centre Europe, France
Alexandre Bouchard-Cote	University of British Columbia, Canada
Abdeslam Boularias	Max Planck Institute, Germany
Darius Braziunas	Kobo, Canada
Emma Brunskill	Carnegie Mellon University, USA
Wray Buntine	NICTA, Canberra, Australia
Tiberio Caetano	NICTA, Canberra, Australia
Joaquin Candela	Facebook, USA
Peter Carbonetto	University of Chicago, USA
Lawrence Carin	Duke, USA
Francois Caron	INRIA Bordeaux, France
Robert Castelo	Universitat Pompeu Fabra, Spain
Hei Chan	National Institute of Informatics, Japan
Laurent Charlin	University of Toronto, Canada
Tianjiao Chu	University of Pittsburgh, USA
Tom Claassen	Radboud University Nijmegen, The Netherlands
Adam Coates	Stanford University, USA
Barry Cobb	Virginia Military Institute, USA
Ira Cohen	Hewlett Packard Labs, USA
Diego Colombo	ETH, Switzerland
Paulo Costa	George Mason University, USA
Fabio Cozman	University of Sao Paolo, Brazil
Mark Craven	University of Wisconsin, Madison, USA
James Cussens	University of York, UK
Cassio de Campos	IDSIA, Switzerland

Luis M. De Campos	University of Granada, Spain
Frank Dellaert	Georgia Tech, USA
Sebastien Destercke	CNRS, France
Marek J. Druzdzel	University of Pittsburgh, USA
David Dunson	Duke University, USA
Jennifer Dy	Northeastern University, USA
Gal Elidan	Hebrew University of Jerusalem, Israel
Helene Fargier	IRIT-CNRS, FRance
M. Julia Flores	University of Castilla—La Mancha (UCLM), Spain
James Foulds	University of California, Irvine, USA
Charless Fowlkes	University of California, Irvine, USA
Minos Garofalakis	Technical University of Crete, Greece
Thomas Gartner	University of Bonn, Germany
Jan Gasthaus	University College London, UK
Joydeep Ghosh	University of Texas, Austin, USA
Mark Girolami	University College London, UK
Amir Globerson	Hebrew University, Israel
Vibhav Gogate	University of Texas at Dallas, USA
Moises Goldszmidt	Microsoft Research, USA
Manuel Gomez-Olmedo	Universidad de Granada, Spain
Christophe Gonzales	LIP6-UPMC, France
Mihajlo Grbovic	Yahoo! Labs, USA
Shengbo Guo	Samsung Information Systems America, USA
Yuhong Guo	Temple University, USA
Maya Gupta	University of Washington, USA
Hannaneh Hajishirzi	University of Washington, USAS
David Heckerman	Microsoft Research, USA
Philipp Hennig	Max Planck Institute, Germany
Ralf Herbrich	Amazon, USA
Jesse Hoey	University of Waterloo, Canada
Antti Honkela	University of Helsinki, Finland
Eric Horvitz	Microsoft Research, USA
William Hsu	Kansas State University, USA
Bert Huang	University of Maryland, USA
David Jensen	University of Massachussetts, USA
Albert Xin Jiang	USC, USA
Nebojsa Jojic	Microsoft Research
Kshitij Judah	Oregon State University
Ece Kamar	Microsoft Research
Ashish Kapoor	Microsoft Research
Kalev Kask	University of California, Irvine, USA
Seyoung Kim	Carnegie Mellon University, USA
Sergey Kirshner	Purdue University, USA
David Knowles	Stanford University, USA
Mikko Koivisto	Helsinki Institute for Information Technology, Finland
Vladimir Kolmogorov	Institute of Science and Technology, Austria
Petri Kontkanen	HIIT/CoSCo, Finland
Kevin Korb	Monash University, Australia
Timo Koski	KTH, Sweden
Wojciech Kotlowski	University of Poznan, Poland
Alex Kulesza	University of Michigan, USA
Branislav Kveton	Technicolor Labs, USA
Jerome Lang	LAMSADE, CNRS/Universite Paris-Dauphine, France
Tobias Lang	FU Berlin, Germany
Hugo Larochelle	Universite de Sherbrooke, Canada

Erik Learned-Miller	University of Massachusetts Amherst, USA
Anthony Lee	University of Warwick, USA
Jan Lemeire	Vrije Universiteit Brussel, Belgium
Philippe Leray	University of Nantes, France
Jennifer Listgarten	Microsoft Research, USA
Qiang Liu	University of California, Irvine, USA
Weiru Liu	Queen's University Belfast, UK
Dan Lizotte	University of Waterloo, Canada
Ben London	University of Maryland, USA
Daniel Lowd	University of Oregon, USA
Peter Lucas	Radboud University Nijmegen, The Netherlands
Michael Lyu	The Chinese University of Hong Kong, Hong Kong
Marloes Maathuis	ETH Zurich, Switzerland
Malik Magdon-Ismail	Rensselear Polytechnic Institute, USA
Subramani mani	University of New Mexico, USA
Vikash Mansinghka	MIT, USA
Radu Marinescu	IBM Research, USA
Benjamin Marlin	University of Massachusetts Amherst, USA
Maria Vanina Martinez	The University of Oxford, UK
Andre Martins	Priberam Labs, Portugal
Jon Mcauliffe	University of California, Berkeley, USA
Andrew McCallum	University of Massachusetts Amherst, USA
Talya Meltzer	Hebrew University of Jerusalem, Israel
Ole Mengshoel	Carnegie Mellon University, USA
Taneli Mielikainen	Nokia Research Center Palo Alto, USA
David Mimno	Princeton University, USA
Thomas Minka	Microsoft Research, UK
Joris Mooij	Radboud University Nijmegen, The Netherlands
Sach Mukherjee	Netherlands Cancer Institute, The Netherlands
Iain Murray	University of Edinburgh, UK
Sriraam Natarajan	Wake Forest University Baptist Medical Center, USA
William S. Noble	University of Washington, USA
Michael Osborne	The University of Oxford, UK
Lars Otten	Google, USA
David Page	Department of Biostatistics, University of Wisconsin, Madison, USA
John Paisley	University of California, Berkeley, USA
Jason Pazis	Duke University, USA
Jose Pena	Linkoping University, Sweden
David M. Pennock	Microsoft, USA
Patrice Perny	LIP6–UPMC, France
Jonas Peters	ETH Zurich, Switzerland
Nathalie Peyrard	INRA, France
Kim-Leng Poh	National University of Singapore, Singapore
Hoifung Poon	Microsoft Research, USA
Leonard Poon	Hong Kong Institute of Education, Hong Kong
Bob Price	PARC, USA
David V. Pynadath	University of Southern California, Institute for Creative Technologies, USA
Yuan (Alan) Qi	Purdue University, USA
Zengchang Qin	Beihang University, China
Erik Quaeghebeur	Ghent University, Belgium
Emmanuel Rachelson	ISAE Supaero, France
Piyush Rai	University of Texas, Austin, USA
Roland Ramsahai	University of Cambridge, UK
Vinayak Rao	Duke University, USA
Silja Renooij	University of Utrecht, The Netherlands

Teemu Roos	Helsinki Institute for Information Technology, Finland
Dan Roy	University of Cambridge, UK
Mike Ruberry	Harvard University, USA
Rafael Rumi	Almeria University, Spain
Brian Ruttenberg	Charles River Analytics, USA
S.V.N. Vishwanathan	Purdue University, USA
Regis Sabbadin	INRA, France
Ruslan Salakhutdinov	University of Toronto, Canada
Antonio Salmeron	Almeria University, Spain
Jose San Pedro	Telefonica, Spain
Scott Sanner	NICTA/Australian National University, Australia
Cristina Savin	Cambridge University, UK
Mark Schmidt	INRIA/ENS, France
Jeff Schneider	Carnegie Mellon University, USA
Matthias Seeger	EPFL, Switzerland
Bart Selman	Cornell University, USA
Ross D. Shachter	Stanford University, USA
Cosma Shalizi	Carnegie Mellon University, USA
Ilya Shpitser	University of Southampton, UK
Gerardo I. Simari	The University of Oxford, UK
Tomas Singliar	Boeing, USA
Jim Smith	The University of Warwick, UK
Le Song	GeorgiaTech, USA
Mark Steyvers	University of California, Irvine, USA
Andreas Stuhlmuller	MIT and Stanford University, USA
L. Enrique Sucar	INAOE, Mexico
Joe Suzuki	Osaka University, Japan
Graham Taylor	University of Guelph, Canada
Joshua Tenenbaum	MIT, USA
Robert E. Tillman	Rotella Capital Management, USA
Volker Tresp	Siemens AG, Germany
Matthias Troffaes	Durham University, UK
Charles Twardy	George Mason University, USA
Raquel Urtasun	Toyota Technological Institute at Chicago, USA
Marco Valtorta	University of South Carolina, USA
Laurens van der Maaten	Delft University of Technology, The Netherlands
Vladimir Vovk	Royal Holloway, University of London, UK
Christopher Wakefield	Duke University, USA
Chong Wang	Princeton University, USA
Hao Wang	University of Southern California, USA
Yi Wang	Institute of High Performance Computing, Singapore
Renata Wassermann	Universidade de Sao Paulo, Brazil
Wim Wiegerinck	Radboud University Nijmegen, The Netherlands
Changhe Yuan	Queens College/City University of New York, USA
Kun Zhang	Max Planck Institute for Intelligent Systems, Germany
Onno Zoeter	Xerox Research Centre Europe, France
Geoff Zweig	Microsoft Research, USA

Additional Reviewers

A. Barnard, University of Wisconsin, Madison, USA
Bryant Chen, UCLA, USA
C. Chen, Australian National University, Australia
Eunice Yuh-Jie Chen, UCLA, USA
Shouyuan Chen, The Chinese University of Hong Kong, Hong Kong
Chen Cheng, The Chinese University of Hong Kong, Hong Kong
Diego Colombo, ETH Zurich, Switzerland
Efrat Egozi-Levi, Hewlett Packard
Guy Feldman, Purdue University, USA
José A. Gámez, University of Castilla–La Mancha, Spain
M. Ben Ishak, University of Tunis/University of Nantes, France
Xueyan Jiang, Ludwig Maximilian University of Munich, Germany
Denis Krompass, Ludwig Maximilian University of Munich, Germany
Jun-young Kwak, University of Southern California, USA
Max Libbrecht, University of Washington, USA
Guang Ling, The Chinese University of Hong Kong, Hong Kong
Miles Lopes, University of California, Berkeley, USA
Wenxiu Ma, University of Washington, USA
Jerome Mengin, IRIT-University of Toulouse, France
Christopher Painter-Wakefield, Duke University, USA
Jason Pazis, Duke University, USA
José M. Puerta, University of Castilla–La Mancha, Spain
Mohamed Sharaf, University of South Carolina, USA
Inma Tur, Universitat Pompeu Fabra, Spain
Bo Waggoner, Harvard University, USA
Haiqin Yang, The Chinese University of Hong Kong, Hong Kong
Lin Yuan, Purdue University, USA

Additional Acknowledgments

A number of other people have made significant contributions towards making UAI 2013 possible. We acknowledge and thank:

- David Heckerman, Fabio Cozman, and Kevin Murphy for general advice and assistance related to UAI organization.
- Thao Hoang Xuan for creating and maintaining the UAI 2013 Web site.
- David Poole for assistance in setting up UAI 2013 email accounts and Website.
- Prakash Shenoy for his help with the financial aspects of the conference.
- Laurent Charlin and Richard Zemel for their help with the Toronto Paper-Matching System.
- Drew Frank and Matt Hoffmann for assistance in generating the Proceedings.
- Martijn De Jongh for help with copyright forms and ArXiv.

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 2013 without charging much higher registration fees.

Microsoft® Research

Artificial Intelligence
www.elsevier.com/locate/artint

amazon.com®

Google™

charles river analytics

TOYOTA
INFOTECHNOLOGY CENTER, U.S.A., INC.

facebook®

IBM Research

Best Paper Awards

Best paper award - sponsored by Microsoft ($1000)
Scalable Matrix-valued Kernel Learning for High-dimensional Nonlinear Multivariate Regression and Granger Causality
Vikas Sindhwani, Ha Quang Minh, Aurelie Lozano

Facebook Best Student Paper (and official Runner up/Honourable mention) ($1000)
Optimization With Parity Constraints: From Binary Codes to Discrete Integration
Stefano Ermon, Carla Gomes, Ashish Sabharwal, Bart Selman

Google Best Student Paper ($1000)
On the Complexity of Strong and Epistemic Credal Networks
Denis Maua, Cassio de Campos, Alessio Benavoli, Alessandro Antonucci

Amazon Best Student Paper ($1000)
Constrained Bayesian Inference for Low Rank Multitask Learning
Oluwasanmi Koyejo, Joydeep Ghosh

Proceedings

Generative Multiple-Instance Learning Models For Quantitative Electromyography

Tameem Adel† **Ruth Urner‡** **Benn Smith*** **Daniel Stashuk†** **Daniel J. Lizotte‡**

†Systems Design Engineering
‡David R. Cheriton School of Computer Science
University of Waterloo, Waterloo, ON, Canada

*Department of Neurology
Mayo Clinic, Scottsdale, AZ, United States

Abstract

We present a comprehensive study of the use of generative modeling approaches for Multiple-Instance Learning (MIL) problems. In MIL a learner receives training instances grouped together into bags with labels for the bags only (which might not be correct for the comprised instances). Our work was motivated by the task of facilitating the diagnosis of neuromuscular disorders using sets of motor unit potential trains (MUPTs) detected within a muscle which can be cast as a MIL problem. Our approach leads to a state-of-the-art solution to the problem of muscle classification. By introducing and analyzing generative models for MIL in a general framework and examining a variety of model structures and components, our work also serves as a methodological guide to modelling MIL tasks. We evaluate our proposed methods both on MUPT datasets and on the MUSK1 dataset, one of the most widely used benchmarks for MIL.

1 Introduction

In Multiple-Instance Learning (MIL), training instances are grouped together in *bags* which have labels. Each *instance* in a bag has a label that may be different from that of the bag, but instance labels are not observed; only the label of the bag is available for learning. The MIL framework was first introduced by Dietterich et al. [1997] for a problem in a medical (pharmaceutical) domain. Their task was to predict the binding properties of molecules, which depend on the shape of the molecule. However, a molecule can take on several shapes. Thus, each molecule is represented as a bag of instances, where each instance represents a shape the molecule can take on. If none of the possible shapes enable binding, the bag (molecule) gets a negative label. But as soon as one shape allows for binding, the bag is labeled positive. The MUSK dataset from this problem has remained one of the most widely used benchmark datasets for MIL tasks.

Following the introduction of the framework, various problems have been expressed as MIL. MIL approaches have, for example, been employed for content-based image retrieval [Maron and Ratan, 1998, Zhang et al., 2002], text classification [Settles et al., 2007, Andrews et al., 2002b], protein identification [Tao et al., 2004], music information retrieval [Mandel and Ellis, 2008] and activity recognition [Stikic and Schiele, 2009]. In medical domains, prediction problems often naturally occur as MIL tasks. One example is the original MUSK prediction task; Dundar et al. [2008] also show that learning problems for computer-aided detection applications can often be considered as MIL problems.

Our work was motivated by an application which uses quantitative analysis of clinically detected electromyographic (EMG) signals to assist with the diagnosis of certain neuromuscular disorders. The diagnosis of a neuromuscular disorder often requires the characterization of several individual muscles. A muscle characterization, in turn, is based on characterizing a sampling of its motor units (MUs). A motor unit potential train (MUPT) created by a MU and extracted from a needle-detected EMG signal can be used to obtain a characterization of the MU (see Section 2 for details). The classification of a muscle based on the set of MUPTs representing a sampling of its MUs can therefore be formulated as a MIL problem wherein each muscle is a bag and each MU of a muscle is an instance of that bag. We propose that *generative* modeling approaches—models that describe the full joint distribution of the data—are useful and effective for data that naturally occurs in MIL form, such as muscle classification based on a set of MUPTs. Predicting with a generative model is particularly suitable for medical domains for several reasons: Generative models allow for expert domain knowledge to be incorporated in an intuitive way, which leads to good inductive bias in the modeling assumptions. As we will demonstrate, a model with good inductive bias (elicited from experts in biomedicine) can result in highly accurate predictions even on the basis of a relatively small training set. Most importantly, they yield not only a classification tool, but a simulation tool for the problem domain. In our setting, such a simulator provides a stepping stone toward a more sophisticated system that not only helps with the diagnosis of neuromuscular disorders but also provides a measure of their severity.

Our contributions are two-fold: First, we provide a state-of-the-art solution to the problem of muscle classification. We show that modeling the muscle as a two-stage generative model (according to the way its MUPTs are actually generated) significantly improves classification accuracy over previous strategies for this task both at the instance and bag level [Adel et al., 2012]. Second, we introduce a general framework and provide intuition and guidelines for applying generative models to MIL problems. Generative models have only recently been successfully applied to MIL tasks [Yang et al., 2009, Foulds and Smyth, 2011]; these can be viewed as special cases of our framework. We compare different possible model structures for MIL generative models both conceptually and experimentally, and we discuss the impact of their differing conditional independence structures and parametric modeling assumptions. We suggest several possible implementations for these structures and validate the proposed methods both on the MUPT and MUSK data. Because we examine a variety of model structures and components (not just those appropriate for muscle classification) our work also serves as a methodological guide to modelling MIL tasks.

2 Muscle Classification using QEMG

Quantitative electromyography (QEMG) is a method used to help diagnose neuromuscular disorders by quantitatively analyzing EMG signals detected during a slight to moderate level, voluntary muscle contraction using an inserted needle electrode. A muscle is comprised of several MUs which are repeatedly activated during muscle contraction. A motor unit consists of a group of muscle fibers and the α motor neuron that activates those fibers. The voltage signal detected by an electrode created by the activation of the fibers of a motor unit is called a motor unit potential (MUP). The train of MUPs created by the repeated activity of a MU is called a MUPT. Each EMG signal is thus a compound signal that represents the sum of the MUPTs of all active motor units. For analysis purposes, an EMG signal is decomposed into

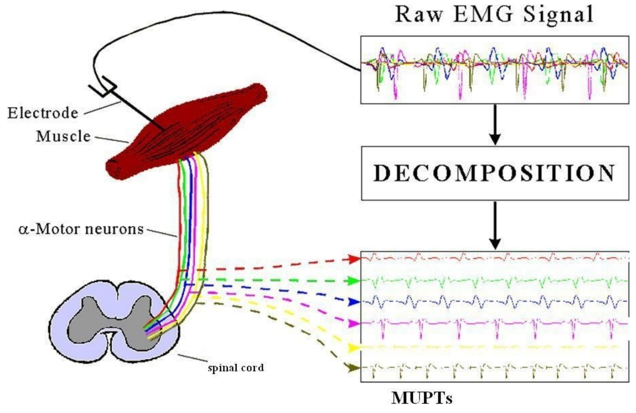


Figure 1: MUPT extraction via EMG Signal Decomposition. Derived from Basmajian and Luca [1985].

its constituent MUPTs using a state-of-the-art pattern recognition based decomposition system (see also the work of Adel et al. [2012] and Farkas et al. [2010] for more details). Figure 1 illustrates this MUPT extraction process. Usually, 4 to 6 EMG signals, each detected with the needle at a distinct location in the muscle, are recorded and decomposed to obtain a representative set of MUPTs of sufficient size (15-25).

Muscles are classified as either *normal*, *myopathic* or *neurogenic* based on clinical expertise. Initially, MUPTs are labeled normal, myopathic or neurogenic on the basis of being detected in a normal, myopathic or neurogenic muscle. For normal muscles, this is largely correct as it is unlikely that a motor unit of a normal muscle would generate a disordered (myopathic or neurogenic) MUPT. However, passing on the muscle label to each MUPT is not accurate for disordered muscles: myopathic and neurogenic muscles commonly have some normal MUs, and thus produce some normal MUPTs. Thus, labelling all MUPTs in a disordered muscle as disordered is incorrect.

Classifying a muscle as normal, myopathic or neurogenic can be posed as a MIL problem in a straightforward manner: In this task, a bag corresponds to the muscle that produces the MUPT instances, with each instance representing a sampled MU within that muscle. The features of an instance correspond to the features used to represent the MUPT of the MU. The instances of a normal bag are all normal, while neurogenic and myopathic bags might contain both normal and neurogenic or myopathic instances, respectively. It is exceedingly unlikely that a neurogenic (resp. myopathic) disordered muscle contains/generates a myopathic (resp. neurogenic) MU/MUPT. While it is possible for a domain expert to manually classify individual MUPTs, this task is time-consuming. Clinical experts therefore typically provide only the diagnosis for the whole muscle, which gives the bag label. Thus a learner must learn how to classify new MUPTs and muscles from a training set providing only muscle labels.

The machine learning techniques implemented in this work are known as *quantitative EMG techniques*. The main goal of quantitative EMG techniques is to extract suitable information from detected EMG signals and then interpret this information to assist with the diagnosis of their respective muscles. It is also desirable that quantitative EMG analysis provides a measure related to the severity of the predicted disorder [Pino, 2008]. Two muscles can both be myopathic, but one may be mildly myopathic while the other is severely myopathic; it is hypothesized that severe cases are indicated by increased numbers of MUPTs from within their class as well as that some of the MUPTs may be outliers within their class. By identifying when an instance is atypical within its class label, in future our generative model may be used for estimating the severity of muscular disorders. Our ultimate goal is to build a clinical decision support system that assists with the diagnosis of muscles (both in terms of classification and assessment of severity) by inspecting sets of MUPTs extracted from EMG signals.

3 Related work

The last decade and a half has seen the development of a large body of work on MIL, both in terms of theoretical analysis and the development of practical algorithms for various application areas as we mentioned in the introduction.

Dietterich et al. [1997] suggest several algorithms for learning axis-aligned rectangles for the original MIL problem on the MUSK data. Maron and Lozano-Pérez [1998] introduced the diverse density (DD) algorithm, a paradigm for MIL that, similar to the axis-aligned rectangle learning approach, assumes that there is a specific region of positive instances to be identified in the feature space. The algorithm has been further developed to EM-DD by Zhang and Goldman [2001]. This is one of the most successful approaches for MIL and we discuss how it relates to our framework in Section 4.2. Wang and Zucker [2000] adapted Nearest Neighbor learning to MIL. Several studies present kernels to use Support Vector Machines on MIL problems [Gärtner et al., 2002, Andrews et al., 2002a, Tao et al., 2004], or adaptations of boosting [Andrews and Hofmann, 2003, Xu and Frank, 2004, Viola et al., 2005] and, more recently, incorporate methods from semi-supervised or active learning into the MIL setting [Rahmani and Goldman, 2006, Zhou and Xu, 2007, Settles et al., 2007]. The original bag labeling rule (where the label of the bag is the logical OR of its instances as in the MUSK data) has been modified and generalized to apply to other areas (Foulds and Frank [2010] provide an overview).

Long and Tan [1998] analyze the original problem of learning axis-aligned rectangles from MIL data in the Probably Approximately Correct (PAC) learning framework. This set-up assumes independence of the instances that occur together in a bag and the goal is to learn a low-error instance level classifier. Blum and Kalai [1998] show that this framework is equivalent to PAC-learning with one-sided noise, a problem that has recently been analyzed in Simon [2012]. However, in most MIL problems, it is not appropriate to assume that instances occurring together in a bag are conditionally independent and the goal is to learn a bag-level classifier rather than an instance-level classifier. Sabato et al. [2010] provides a comprehensive study with upper and lower bounds on the sample complexity of the bag-level learning problem without the independence assumption. Diochnos et al. [2012] tighten some of those lower bounds.

Generative model approaches have only rather recently been introduced to the MIL setting [de Freitas and Kück, 2005, Yang et al., 2009, Foulds and Smyth, 2011]. The former two studies suggest more complex model structures for modifications of the MIL problem. The work of Foulds and Smyth [2011] fits into our framework with specific choices for the model components. We discuss the modeling choices that were made in Foulds and Smyth [2011] and Yang et al. [2009] in the context of our framework below.

4 Generative Models for MIL

We denote random variables in upper case, and their realizations in lower case. Let t be the number of possible bag/instance labels. Let $B \in \{1, 2, ..., t\}$ represent a bag's label, and let $I_j \in \{1, 2, ..., t\}$ represent the label of the jth instance belonging to the bag. The number of instances in the bag is denoted by m; we refer to the m instance labels together as the vector $\vec{I}$. Let $\vec{F}_j \in \mathbb{R}^p$ be the p-dimensional feature vector belonging to the jth instance. We index elements of a vector with a square-bracketed subscript, so $\vec{F}_{j[k]}$ is the kth element of the observed feature vector of the jth instance in a bag. In our models, marginal and conditional distributions involving only I_j and $\vec{F}_j$ are the same for all j, so we refer to a "generic" instance label as I and a "generic" feature vector as $\vec{F}$.

Most MIL work to date considers binary labels, i.e. $t = 2$. In our muscle classification problem $t = 3$, since a bag or instance can be either *normal* ($B = 1$), *myopathic* ($B = 2$), or *neurogenic* ($B = 3$). Fur-

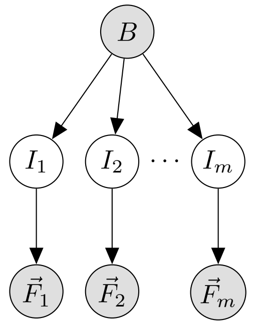


Figure 2: The BIF Model Structure. Parameters for $P(I_j|B)$ and for $P(\vec{F}_j|I_j)$ are tied across j.

thermore, in our problem, a bag may only generate a *compatible* instance label: We call the value of a single instance label i_j *compatible* with a bag label b iff $i_j \in \{1\} \cup \{b\}$. We call a joint labelling $\vec{i} = i_1, ..., i_m$ *feasible* iff $\exists b(\forall j, i_j \in \{1\} \cup \{b\})$.

We begin by discussing possible Bayes net structures for our MIL model in Section 4.1. We then discuss possible modelling choices for marginal and conditional distributions in Section 4.2.

4.1 Model Structures

The main inductive bias that we retain from the original MIL formulation is the assumption that the bag label is conditionally independent of the feature vectors given their corresponding instance labels. This is implied by the assumption that the feature distribution of normal instances in a normal bag should be the same as the feature distribution of normal instances in an abnormal bag. This restricts us to three possible Bayes net structures presented below, two of which we will use as candidate structures for our generative model. For completeness, we also discuss a fourth structure that does not satisfy the conditional independence assumption. We name the structures based on the partial order in which the variables appear in the graph. Alpaydin [2010] gives a concise overview of directed graphical models; Koller and Friedman [2009] give a comprehensive treatment.

4.1.1 BIF: $B \rightarrow \boxed{I \rightarrow \vec{F}\ \ _m}$

This structure best represents the generative process underlying our MUPT data. Under this structure, the bag (muscle) generates its m instances (MUPTs) independently given its label. Each instance in turn generates its own feature vector given *its* label, but independent of the bag label and independent of the other instances and features in the bag. The structure of this model is given concisely by the plate diagram $B \rightarrow \boxed{I \rightarrow \vec{F}\ \ _m}$ for which we give the expanded version in Figure 2. The other model structures we consider are the same except for the directions of the edges. We will see that the choice of the edge directions has important consequences.

For this structure, we must learn $P(B)$ from observed data, and learn $P(I|B)$ and $P(\vec{F}|I)$ using a hidden variable method like EM. Constraints on $P(I|B)$ are simple to encode; to ensure that a bag can *never* generate an incompatible instance, we can restrict the values in the conditional probability table of $I|B$ by requiring $P(I = i|B = b) = 0$ for all $i \notin \{1\} \cup \{b\}$. It follows that sets of instance labels drawn from this model are always compatible with the bag label. Furthermore, we can easily impose a Dirichlet prior on the proportion of instance labels that match the bag label while obeying these constraints. Since we have continuous features, $P(\vec{F}|I)$ is modeled using density estimation.

The main departure this model makes from other probabilistic models for MIL is that it assumes that the bag label is the cause of the instance label, rather than the other way around. Under this model, it is possible for a non-normal bag to produce all normal instances, which is disallowed in other MIL models. However, for our system, this is entirely appropriate; it is possible (though unlikely) for a muscle with a myopathic or neurogenic disorder to produce all normal MUPTs. Among existing models, that of Yang et al. [2009] is most similar to BIF.

4.1.2 FIB: $B \leftarrow \boxed{I \leftarrow \vec{F}\ \ _m}$

FIB represents another way of expressing an MIL model where the instances generate the bag label. Under this structure, the feature vectors are drawn from some $P(\vec{F})$, they then generate the instance labels, which in turn determine the bag label. The probability $P(I|\vec{F})$ can be expressed using any discriminative learner, which is attractive, though we still must use a hidden variable method like EM for training because the instance labels are not observed. In order to make the model fully generative we must also model $P(\vec{F})$ using density estimation. In previous work, for example, EM-DD, this model structure is used (though not made explicit) since if the model does *not* need to be generative (i.e. if we will always condition on $\vec{F}$) then density estimation is not required at all. We will show in Section 4.2 that the well-known EM-DD MIL algorithm [Han et al., 2007] can be implemented using this model structure.

Note that B has all m instances as parents, and m can vary from muscle to muscle, so we must express $P(B|\vec{I})$ to allow different sized joint labellings; this is discussed in Section 4.2. Unfortunately, in our 3-class setting, this structure suffers from an important drawback: it offers no way of prohibiting infeasible instance label assignments, i.e. assignments where for example $I_1 = 2$ and $I_2 = 3$. In order to have a fully consistent generative FIB model, therefore, we must add an additional possible bag "label" $b = 0$ that has positive probability given infeasible labelings. This does not reflect the generative process of the MUPT data, but we can still use this model structure and condition on the event $B \neq 0$ where necessary. Foulds and Smyth [2011] note that prior knowledge about the frequency of instance labels given bag labels is difficult to incorporate with a directed edge from I to B.

4.1.3 IBF: $B \leftarrow \boxed{I \rightarrow \vec{F} \;\; _m}$

Under this structure, the instances are generated independently according to some $P(I)$, and they subsequently generate both the bag label and the feature vectors. This model structure is essentially the "Multi-Instance Mixture Model" of Foulds and Smyth [2011]. As in the FIB model, we have m directed edges from the I_j to B, which causes the same drawbacks described in the FIB model but does *not* give us the additional flexibility of using a discriminative learner for the instance labels. Therefore, we will not consider this structure for our generative model. Foulds and Smyth [2011] observe that the EM-DD algorithm can be expressed using this structure given an appropriate model of $P(\vec{F}|I)$ and a "discriminative learning objective"; this is equivalent to our FIB structure described above.

4.1.4 Alternative model BFI: $B \rightarrow \boxed{I \leftarrow \vec{F} \;\; _m}$

This model attempts to combine two attractive properties: The ability to use a discriminative model $P(I|\vec{F})$, and the ability to easily assign values for $P(I|B)$. Unfortunately, in this model, B and $\vec{F}$ are dependent given I, which we know to be untrue in our problem. In this model, normal instances where $B = 1$ may have a different feature distribution from normal instances where $B = 2$. Since we count on being able to generalize from normal instances in normal bags to normal instances in abnormal bags, this model is not appropriate. Note that the BFI model is essentially a clustering model with I as the cluster label and B acting simply as an additional feature along with $\vec{F}$.

4.2 Model Components

Choosing the model structure determines the conditional independence properties of our model but does not specify a form for the various distributions. We discuss some possibilities for components of the model that have different assumptions and inductive biases.

4.2.1 $P(B)$ and $P(I|B)$ for the BIF Structure

Since B and I take on a small number of discrete values, a tabular representation is appropriate. As noted earlier, one can impose restrictions on possible values of $I|B$ by clamping appropriate values in the conditional probability table.

4.2.2 $P(\vec{F}|I)$ for the BIF Structure

Because we assume a continuous feature space, $P(\vec{F}|I)$ can be modeled using any density estimation method. We discuss some well-known possibilities here.

Multivariate Gaussian One simple choice for $P(\vec{F}|I = i)$ is a multivariate Gaussian distribution with mean μ_i and covariance Σ_i for $i \in \{1, ..., t\}$. Depending on the availability of data and desired modelling assumptions, one can restrict Σ_i to be diagonal.

Gaussian Copula with KDE Marginals A drawback of the multivariate Gaussian approach is that much real-world data is not in fact Gaussian. Since we want our generative model to be as realistic as possible, we propose a copula-based model that is practical to estimate and can fit the observed data more closely.

Sklar's theorem [1959] implies that any multivariate density g with marginal densities $g_1, g_2, ..., g_p$ and marginal cumulative distribution functions (CDFs) $G_1, G_2, ..., G_p$ can be expressed in the form

$$g(\vec{f}) = g_1(\vec{f}_{[1]}) \cdot g_2(\vec{f}_{[2]}) \cdot ... \cdot g_p(\vec{f}_{[p]}) \cdot c(G_1(\vec{f}_{[1]}), G_2(\vec{f}_{[2]}), ..., G_p(\vec{f}_{[p]})) \quad (1)$$

where c is a *copula density* that captures the dependence structure of the feature vector $\vec{F} = (\vec{F}_{[1]}, ..., \vec{F}_{[p]})$. If the elements of $\vec{F}$ are independent, then $c \equiv 1$.

Because they are all one-dimensional, the marginals can be estimated well using Kernel Density Estimation (KDE) Alpaydin [2010] even with a modest amount of data, giving $\hat{g}_k$ and $\hat{G}_k$ for $k = 1...p$. This allows our model to capture non-Gaussian aspects of the data, such as relatively heavy or light tails, skewness, or even multi-modality, thus making it more realistic.

The copula model allows us to achieve more high-fidelity marginals without resorting to an unrealistic independence assumption: we can still capture pairwise dependencies in the data by assuming a parametric form for c and estimating the necessary parameters. We will assume a Gaussian copula, whose parameter is the covariance matrix of $(\Phi^{-1}(G_1(\vec{F}_{[1]})), \Phi^{-1}(G_2(\vec{F}_{[2]})), ..., \Phi^{-1}(G_p(\vec{F}_{[p]})))$ where Φ^{-1} is the inverse of the standard normal CDF.

This can be estimated by the empirical covariance of $(\Phi^{-1}(\hat{G}_1(\vec{F}_{[1]})), \Phi^{-1}(\hat{G}_2(\vec{F}_{[2]})), ..., \Phi^{-1}(\hat{G}_p(\vec{F}_{[p]})))$ over the observed $\vec{f}|I = i$ in the data. (We estimate a separate copula model for each possible value of I.) Other, more flexible copula models are possible; we have elected to use the Gaussian copula for simplicity.

Kernel Density Estimation Kernel Density Estimation (KDE) is a non-parametric method that estimates a probability density or distribution function by summing up *kernel* functions placed at every observed data point. We use the most common form of KDE, which uses a Gaussian kernel. To choose the kernel width, we employ the *maximum smoothing principle* [Terrell, 1990] as a simple but effective choice; other more fine-tuned choices are possible. The advantage of KDE is that it is capable of modeling complex marginals *and* complex dependencies among the variables of interest, but it does not always work well in moderate to high dimensions.

4.2.3 $P(\vec{F})$ for the FIB Structure

In principle, any of the density estimators proposed for $P(\vec{F}|I)$ could be used here; however, the marginal $P(\vec{F})$ is likely to be multi-modal, so the copula or KDE models may be more appropriate.

4.2.4 $P(I|\vec{F})$ for the FIB Structure

Since I has a discrete domain, any classification method that supplies class probabilities can be used to model $P(I|\vec{F})$. We examine four such methods.

Logistic Regression This well-known model assumes $P(I = i|\vec{F}) \propto \exp^{\vec{\beta}_{i[0]} + \vec{\beta}_{i[1:p]}^{\top} \vec{f}}, i < t$. Note that the maximum likelihood estimate of β is not unique if the data are linearly separable.

Support Vector Machines Although the classic SVM formulation [Cortes and Vapnik, 1995] does not provide conditional class probabilities, subsequent work [Huang et al., 2006, Chang and Lin, 2011] has added this capability. It has the added advantage that in the event of feature separability, we get a large-margin classifier whereas logistic regression would fail to converge. Furthermore, kernelized SVMs allow us to easily create non-linear separators in a feature space.

K Nearest Neighbours If the decision boundary between instance labels is believed to be complex and if we have sufficient data, a non-parametric model may be warranted. K nearest neighbours uses the empirical distribution of the instance labels of the K closest feature vectors to $\vec{f}$ to estimate $P(I|\vec{F} = \vec{f})$.

"Diverse Density" When bag and feature labels are binary ($t = 2$ where 2 is positive and 1 is negative) we may assume $P(I = 2|\vec{F}) = \exp(-\sum_{k=1}^{p} s_{[k]}^2 (\vec{f}_{[k]} - \vec{w}_{[k]})^2)$. Here, s and $\vec{w}$ are parameters fit by maximum likelihood. Note that this is *not* a gaussian distribution; its conditional distributions are Bernoulli such that $P(I = 2|\vec{F} = \vec{f}) \approx 1$ when $\vec{f}$ is near $\vec{w}$. We infer that the reason for its use in the DD and EM-DD algorithms comes from the assumption that the positive instances were localized in feature space, whereas the negative instances were assumed only to be far from the positive ones; they were explicitly assumed not to form a cluster of their own. Because it requires $t = 2$ we cannot use this model for our MUPT data, but we can use it on the MUSK data (described later) for comparison purposes.

4.2.5 $P(B|I)$ for the IBF and FIB Structures

Since m varies from bag to bag, we must express $P(B|I)$ as a function that can take a variable number of parameters. Recall that in this setting, we must allow for the possibility that the joint labelling of $\vec{I}$ is not feasible; we add $b = 0$ to the domain of B to capture this event. We can adhere to the standard MIL assumptions by making $P(B|\vec{I})$ deterministic as follows. For feasible labelings, we set

$$P(B = b|I_1, I_2, ..., I_m) = \begin{cases} 1 & \text{if } b = \max_j I_j \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

and for infeasible labelings we set

$$P(B = b|I_1, I_2, ..., I_m) = \begin{cases} 1 & \text{if } b = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

5 Learning and Inference

All of the components described in Section 4.2 have associated off-the-shelf learning algorithms for completely observed data. We must learn our models without ever observing I (though with substantial information about I provided through B), so we use a hard-Expectation-Maximization (EM) procedure for simultaneously learning the model parameters and inferring the most likely I given the observed B and $\vec{F}$. This worked well on our MUPT data; the conceptual groundwork we lay here could also be used with sampling-based approaches if desired.

5.1 Learning

For learning, we use a "hard-EM" approach [Koller and Friedman, 2009]. We assume access to a collection of n bags of the form $(b, \vec{f}_1, \vec{f}_2, ..., \vec{f}_{m_\nu})_\nu, \nu \in$

$\{1, ..., n\}$ which are independent and identically distributed. Given an initial label assignment to all of the instances in our dataset, our learning method has a straightforward implementation; a sketch is presented in Algorithm 1. We discuss the two main steps.

5.1.1 Parameter Estimation

BIF For the BIF model, we must estimate $P(B)$, $P(\vec{I}|B)$, and $P(\vec{F}|I)$. The marginal probability $P(B)$ is estimated from observed bag label counts only; it does not change across iterations. Because we assume $P(I|B)$ is the same for all instances in all bags, we pool all the bags together and use the aggregated counts to estimate $P(I|B)$. We may add "pseudo-counts" to this estimate if a dirichlet prior is desired; in our experiments we assume for each bag type that we have seen each compatible instance label once, and each incompatible label zero times. To learn $P(\vec{F}|I)$, again we may pool all of the instances together to learn t density estimates $P(\vec{F}|I = 1), P(\vec{F}|I = 2), ..., P(\vec{F}|I = t)$.

FIB For the FIB model, we must estimate $P(\vec{F})$ and $P(I|\vec{F})$; we assume that $P(B|I)$ is fixed according to the standard MIL definition. To estimate $P(\vec{F})$, we pool all feature vectors together and estimate the necessary parameters. These are completely observed so $P(\vec{F})$ does not change across iterations. To learn $P(I|\vec{F})$, again we pool all of the instances together to learn the conditional distribution using a supervised learning method.

5.1.2 Label Updating

To update the labels for each bag given the learned parameters, we must find the most likely instance labels $\vec{i}$ given the observed data, that is, we must compute $\text{argmax}_{\vec{i}} P(\vec{I} = \vec{i}|B = b, \vec{F}_1 = \vec{f}_1, ..., \vec{F}_m = \vec{f}_m)$ for each bag.

BIF From the conditional independence structure of the BIF model, we have

$$P(\vec{I} = \vec{i}|B = b, \vec{F}_1 = \vec{f}_1, ..., \vec{F}_m = \vec{f}_m) \propto P(\vec{I} = \vec{i}|B = b)P(\vec{F}_1 = \vec{f}_1, ..., \vec{F}_m = \vec{f}_m|\vec{I} = \vec{i}).$$

Since the labels and feature vectors for different instances are independent given the bag label, we have

$$P(\vec{I} = \vec{i}|B = b)P(\vec{F}_1 = \vec{f}_1, ..., \vec{F}_m = \vec{f}_m|\vec{I} = \vec{i}) = \prod_{j=1}^{m} P(I_j = i_j|B = b)P(\vec{F}_j = \vec{f}_j|I_j = i_j),$$

so to maximize the probability of the joint label assignment $\vec{i}$, we may maximize each instance label in-

Algorithm 1 Hard EM Algorithm Sketch

```
for all bags do {initialize instance labels}
    $\vec{i} \leftarrow b$
end for
repeat
    learn model components {M-step}
    for all bags do {relabel instances: E-step}
        $\vec{i} \leftarrow \text{argmax}_{\vec{i}} P(\vec{I} = \vec{i}|B = b, \vec{F}_1 = \vec{f}_1, ..., \vec{F}_m = \vec{f}_m)$
    end for
until instance labels do not change
```

dependently:

$$\vec{i}^*_{[j]} \leftarrow \underset{i_j \in \{1,...,t\}}{\text{argmax}} P(I_j = i_j|B = b)P(\vec{F}_j = \vec{f}_j|I_j = i_j).$$

FIB From the conditional independence structure of the FIB model, we have

$$P(\vec{I} = \vec{i}|B = b, \vec{F}_1 = \vec{f}_1, ..., \vec{F}_m = \vec{f}_m) \propto P(B = b|\vec{I} = \vec{i})P(\vec{I} = \vec{i}|\vec{F}_1 = \vec{f}_1, ..., \vec{F}_m = \vec{f}_m).$$

However, in this model, the instance labels are *not* conditionally independent given the bag label and we cannot maximize them independently. For example, if $B = 2$, and $I_1, I_2, ..., I_{m-1}$ are all equal to 1, then I_m must be equal to 2 according to the MIL assumption encoded in $P(B|\vec{I})$. However, we can still avoid searching over all t^m possible label vectors. We defined in Section 4 that if $\vec{i}$ is a feasible vector for $B = b$, then we have $P(B = b|\vec{i}) = 1$ and therefore we have

$$P(B = b|\vec{i})P(\vec{I} = \vec{i}|\vec{F} = \vec{f}) = \prod_{j=1}^{m} P(I_j = i_j|\vec{F}_j = \vec{f}_j).$$

We can find the best feasible $\vec{i}$ in two steps:

1. Let $\vec{i}^*_{[j]} \leftarrow \text{argmax}_{i_j \in \{1\} \cup \{b\}} P(I_j = i_j|\vec{F}_j = \vec{f}_j)$
2. If $\vec{i}^* = \vec{1}$, set $\vec{i}^*_{[k^*]} \leftarrow b$ for k^* given by $\underset{k}{\text{argmax}} \left[P(I_k = b|\vec{F}_j = \vec{f}_j) \prod_{j \neq k} P(I_j = 1|\vec{F}_j = \vec{f}_j)\right]$.

The vector $\vec{i}^*$ is feasible and maximizes $\prod_{j=1}^{m} P(I_j = i_j|\vec{F}_j = \vec{f}_j)$ over all feasible vectors when $B = b$.

5.2 Inference

Once we have learned all of the model parameters, given a new bag where only feature values $\vec{f}$ are observed, we wish to compute $\text{argmax}_{\vec{i},b} P(I = \vec{i}, B = b|\vec{F}_1 = \vec{f}_1, ..., \vec{F}_m = \vec{f}_m)$. These are the most likely bag and instance labels given the m feature vectors in the new bag.

BIF In the BIF model,

$$\begin{aligned}
&\operatorname*{argmax}_{\vec{i},b} P(\vec{I}=\vec{i}, B=b|\vec{F}_1=\vec{f}_1,...,\vec{F}_m=\vec{f}_m)\\
&=\operatorname*{argmax}_{\vec{i},b} P(B=b)P(\vec{I}=\vec{i}|B=b)\\
&\qquad\qquad \cdot P(\vec{F}_1=\vec{f}_1,...,\vec{F}_m=\vec{f}_m|\vec{I}=\vec{i})\\
&=\operatorname*{argmax}_{b}\Big[P(B=b)\operatorname*{argmax}_{\vec{i}}\Big(P(\vec{I}=\vec{i}|B=b)\\
&\qquad\qquad \cdot P(\vec{F}_1=\vec{f}_1,...,\vec{F}_m=\vec{f}_m|\vec{I}=\vec{i})\Big)\Big].
\end{aligned}$$

Therefore we can apply the instance label updating method presented in Section 5.1 for each possible bag label and weight them according to $P(B)$ to find the joint MAP assignment to b and $\vec{i}$.

FIB In the FIB model,

$$\begin{aligned}
&\operatorname*{argmax}_{\vec{i},b} P(\vec{I}=\vec{i}, B=b|\vec{F}_1=\vec{f}_1,...,\vec{F}_m=\vec{f}_m)\\
&=\operatorname*{argmax}_{b}\Big[\operatorname*{argmax}_{\vec{i}}\Big(P(B=b|\vec{I}=\vec{i})\\
&\qquad\qquad \cdot P(\vec{I}=\vec{i}|\vec{F}_1=\vec{f}_1,...,\vec{F}_m=\vec{f}_m)\Big)\Big]
\end{aligned}$$

Therefore we can apply the instance label updating method presented in Section 5.1 for each possible bag label to find the joint MAP assignment to b and $\vec{i}$.

6 Experiments

We now give the details of how our MUPT dataset was constructed, and we discuss the results of the various generative models as applied to our MUPT dataset. We find that the BIF structured models perform very well for several different component choices. The FIB structured models perform less well, but still much better than chance on both bags and instances. Based on our results, we recommend structure and component choices that lead to a high-fidelity generative model of MUPT data. We also give the performance of the models on the MUSK1 dataset [Dietterich et al., 1997].

6.1 The MUPT dataset

Recall that each detected EMG signal is a composite signal that represents the activity of all of the MUs that were active during a muscle contraction. After acquiring an EMG signal, it is decomposed into its constituent MUPTs, each of which ideally represents the electrical activity of a single, sampled MU. As such, the MUPTs are our instances and are the source of our instance level features. From each MUPT, it is common practice to compute a *MUP template* which is a single MUP whose shape is representative of all MUPs in the MUPT [Stashuk, 1999]. All but two of the features we use are functions of this MUP template, while the rest of the features describe aspects of the MUPT itself. We use $p = 8$ features, which were chosen by automated feature selection (both wrapper- and filter-based) in prior work [Adel et al., 2012].

1. **Number of turns** is the number of positive and negative peaks; a function of the MUP template.
2. **Amplitude** represents the maximum difference of voltage between two points [Dumitru et al., 1995]; a function of the MUP template.
3. **Area** represents the area under the curve; a function of the MUP template.
4. **Thickness** refers to the ratio of area to amplitude; a function of the MUP template.
5. **Size index** given by $2\log(\text{amplitude}) + \frac{\text{area}}{\text{amplitude}}$; a function of the MUP template.
6. **Turn width** is given by $\frac{\text{duration}}{\text{turns}}$. Duration is the interval from the first signal deflection from baseline to its final return to baseline [Dumitru et al., 1995]; a function of the MUP template.
7. **Firing rate MCD (mean consecutive difference)** refers to the sequential change in the firing rate of the MU over time; a function of the MUPT.
8. **A-jiggle** is a measure of the shape variability of band-pass filtered MUPs (2nd derivative of the signal); a function of the MUPT.

We have two MUPT datasets, one acquired from upper-leg recordings containing 88 bags and 1534 instances, and another acquired from lower-leg recordings with 70 bags and 1500 instances. All data were collected under IRB approval and and were deidentified. Prior versions of the MUPT data were used by Adel et al. [2012]; our versions have been cleaned to remove obvious outlier errors. For example, instances with highly improbable feature values were removed.

6.2 Results

Table 1 shows the performance of different models on our data. Because one of the authors [TA] manually labeled the instances, we can estimate both the accuracy of each model for classifying bags *and* the accuracy for classifying instances given only the features within a new bag. Note, the manually assigned instance labels were *not* used for learning or inference. The accuracy results were computed using leave-one-bag-out cross-validation. We also present the log likelihood of the observed data maximized over the model parameters and the hidden instance labels, which measures how well the models fit the training data.

We present the results for the BIF structure using five different density estimators for $P(\vec{F}|I)$. We use two

Table 1: MUPT Dataset Results. To give a sense of the statistical uncertainty, we mark all accuracies that are within the 99% Bernoulli confidence interval of the maximum observed accuracy in bold. We mark the highest log likelihoods for the BIF and FIB structures in italics.

Upper Leg	BIF: $B \rightarrow \boxed{I \rightarrow \vec{F}}_m$					FIB: $B \leftarrow \boxed{I \leftarrow \vec{F}}_m$				Non-MIL
Rnd: 0.33	$\perp\!\!\!\perp$Gauss	$\perp\!\!\!\perp$Cop.	Gauss	Cop.	KDE	LR	KNN	SVM	QDA	$\perp\!\!\!\perp$QDA
Bag Acc.	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	0.728	0.568	0.250	**0.898**	0.841
Inst. Acc.	**0.984**	**0.978**	**0.983**	**0.980**	**0.983**	0.728	0.674	0.415	**0.978**	0.850
Log lik.	-36843	-37104	-36810	-37066	*-34726*	*-32889*	-32998	-34382	-32938	—

Lower Leg	BIF: $B \rightarrow \boxed{I \rightarrow \vec{F}}_m$					FIB: $B \leftarrow \boxed{I \leftarrow \vec{F}}_m$				Non-MIL
Rnd: 0.33	$\perp\!\!\!\perp$Gauss	$\perp\!\!\!\perp$Cop.	Gauss	Cop.	KDE	LR	KNN	SVM	QDA	$\perp\!\!\!\perp$QDA
Bag Acc.	**0.986**	**0.971**	**0.971**	**0.957**	0.886	0.814	0.586	0.371	0.886	0.771
Inst. Acc.	**0.946**	**0.899**	**0.931**	0.880	0.859	0.543	0.571	0.469	0.915	0.781
Log lik.	-38035	-38206	-37980	-38141	*-35999*	*-34833*	-34952	-35598	-35128	—

Table 2: MUSK1 Dataset Results

MUSK1	BIF: $B \rightarrow \boxed{I \rightarrow \vec{F}}_m$					FIB: $B \leftarrow \boxed{I \leftarrow \vec{F}}_m$					Non-MIL
Rnd: 0.50	$\perp\!\!\!\perp$ Gauss	$\perp\!\!\!\perp$ Cop.	Gauss	Cop.	KDE	LR	KNN	SVM	QDA	DD	QDA
Bag Acc.	**0.870**	**0.848**	0.696	0.641	**0.772**	**0.783**	**0.772**	**0.837**	**0.837**	0.620	**.783**
Log lik.	*-14921*	-18437	-45815	-51031	-33591	-34086	-34120	-34076	*-34056*	-34105	—

versions each of the Gaussian and copula models, one assuming independence between elements of the feature vectors given the instance labels (e.g. a diagonal covariance matrix) indicated by the prefix $\perp\!\!\!\perp$, and one assuming pairwise correlations. The marginals of the copula models are estimated using KDE with a Gaussian kernel and the maximum smoothing principle (MSP) bandwidth [Terrell, 1990]. We also give results for a multi-dimensional KDE for $P(\vec{F}|I)$, again with the MSP bandwidth.

We present results for the FIB structure using four different discriminative learning models. In all cases, $P(\vec{F})$ was estimated using a multi-dimensional KDE with the MSP bandwidth. The discriminative learners were Logistic Regression (LR), K-nearest neighbors with $K = 7$ (KNN), SVMs with a radial basis function kernel, $C = 1$ and $\gamma = 1/8$, and Quadratic Discriminant Analysis (QDA). The parameter K was chosen based on past experience with the data; SVM parameters are defaults. In the last column, we also present results using Quadratic Discriminant Analysis in a *non* multiple-instance setting by assuming the instance labels are in fact the bag labels and labelling new bags by majority vote.

Table 2 shows results on the MUSK1 dataset, which contains 92 bags and 476 instances. We use the same models and add a version of the FIB model with the "Diverse Density" (DD) model for $P(I|\vec{F})$. Since the data are 166-dimensional, as a pre-processing step, we use PCA to eliminate near-collinearity; we choose enough components to capture 90% of the variance, leaving us with $p = 76$ features. Results are not state-of-the-art—Zhang and Goldman [2001] achieve 96.8%—but moderately good; among our models the BIF model with independent Gaussians for $P(\vec{F}|I)$ has the highest cross-validation accuracy and log likelihood. No expert instance labels exist for MUSK1.

7 Conclusions

Results on the MUPT data indicate that all of our BIF-based generative models perform better than previous state-of-the-art work by Adel et al. [2012], whose best leave-one-bag-out bag label accuracy was 82.3% (lower leg.) In addition, we demonstrate that we are able to recover the instance labels with very high accuracy. The FIB models had worse performance on the MUPT data but better on the MUSK1 data, suggesting they may be useful for other tasks. If muscle classification accuracy is paramount, the parametric model components (Gaussian and Copula) appear best, but if high-fidelity simulation is paramount, then the KDE model component is a better fit to the observed data.

We have introduced a general framework for generative models in MIL . Although MIL is a well-developed sub-field of Machine Learning, generative model approaches had not received much attention so far. Our results suggest that models that are well aligned with the actual data generation in a problem domain (the BIF structure in the case of our muscle classification task) are an excellent choice for classification and modeling purposes.

References

T. Adel, B. Smith, and D. Stashuk. Muscle categorization using pdf estimation and naïve Bayes classification. In *IEEE Engineering in Medicine & Biology Society (EMBC)*, pp. 2619–22, 2012.

E. Alpaydin. *Introduction to Machine Learning.* MIT Press, 2nd edition, 2010.

S. Andrews and T. Hofmann. Multiple-instance learning via disjunctive programming boosting. In *NIPS*, pp. 65–72, 2003.

S. Andrews, T. Hofmann, and I. Tsochantaridis. Multiple instance learning with generalized support vector machines. In *AAAI/IAAI*, pp. 943–944, 2002a.

S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *NIPS*, pp. 561–568, 2002b.

J. Basmajian and C. D. Luca. *Muscles Alive: Their Functions Revealed by Electromyography.* Williams & Wilkins, 1985.

A. Blum and A. Kalai. A note on learning from multiple-instance examples. *Mach. Learn.*, 30(1):23–29, 1998.

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.

C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, Sept. 1995.

N. de Freitas and H. Kück. Learning about individuals from group statistics. In *UAI*, pp. 332–339, 2005.

T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artif. Intell.*, 89(1-2):31–71, 1997.

D. I. Diochnos, R. H. Sloan, and G. Turán. On multiple-instance learning of halfspaces. *Inf. Process. Lett.*, 112 (23):933–936, 2012.

D. Dumitru, A. Amato, and M. Zwarts. *Electrodiagnostic Medicine.* Hanley & Belfus, first edition, 1995.

M. Dundar, G. Fung, B. Krishnapuram, and R. B. Rao. Multiple-instance learning algorithms for computer-aided detection. *IEEE Trans. Biomed. Engineering*, 55 (3):1015–1021, 2008.

C. Farkas, D. Stashuk, A. Hamilton-Wright, and H. Parsaei. A review of clinical quantitative electromyography. *Crit. Rev. Biomed. Eng*, 38(5):467–485, 2010.

J. R. Foulds and E. Frank. A review of multi-instance learning assumptions. *Knowledge Eng. Review*, 25(1): 1–25, 2010.

J. R. Foulds and P. Smyth. Multi-instance mixture models. In *SDM*, pp. 606–617. SIAM, 2011.

T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *ICML*, pp. 179–186, 2002.

F. Han, D. Wang, and X. Liao. An improved multiple-instance learning algorithm. In *Advances in Neural Networks ISNN 2007*, number 4491 in LNCS, pp. 1104–1109, Jan. 2007.

T.-K. Huang, R.-C. Weng, and C.-J. Lin. Generalized Bradley-Terry models and multi-class probability estimates. *J. Mach. Learn. Res.*, 7:85–115, Dec. 2006.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, 2009.

P. M. Long and L. Tan. PAC learning axis-aligned rectangles with respect to product distributions from multiple-instance examples. *Mach. Learn.*, 30(1):7–21, 1998.

M. I. Mandel and D. P. W. Ellis. Multiple-instance learning for music information retrieval. In *ISMIR*, pp. 577–582, 2008.

O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. In *NIPS*, pp. 570–576, 1998.

O. Maron and A. L. Ratan. Multiple-instance learning for natural scene classification. In *ICML*, pp. 341–349, 1998.

L. Pino. Neuromuscular clinical decision support using motor unit potentials characterized by pattern discovery. *University of Waterloo, PhD Thesis*, 2008.

R. Rahmani and S. A. Goldman. Missl: multiple-instance semi-supervised learning. In *ICML*, pp. 705–712, 2006.

S. Sabato, N. Srebro, and N. Tishby. Reducing label complexity by learning from bags. *Journal of Machine Learning Research*, 9:685–692, 2010.

B. Settles, M. Craven, and S. Ray. Multiple-instance active learning. In *NIPS*, pp. 1289–1296, 2007.

H. U. Simon. PAC-learning in the presence of one-sided classification noise. In *ISAIM*, 2012.

A. Sklar. Fonctions de répartition à n dimensions et leurs marges. *Publications de l'Institut de Statistique de l'Université de Paris*, 8:229–231, 1959.

D. Stashuk. Decomposition and quantitative analysis of clinical electromyographic signals. *Med Eng Phys*, 21: 389–404, 1999.

M. Stikic and B. Schiele. Activity recognition from sparsely labeled data using multi-instance learning. In *LoCA*, pp. 156–173, 2009.

Q. Tao, S. D. Scott, N. V. Vinodchandran, and T. T. Osugi. SVM-based generalized multiple-instance learning via approximate box counting. In *ICML*, 2004.

G. R. Terrell. The maximal smoothing principle in density estimation. *Journal of the American Statistical Association*, 85(410):470–477, 1990.

P. A. Viola, J. C. Platt, and C. Zhang. Multiple instance boosting for object detection. In *NIPS*, pp. 1417–1426, 2005.

J. Wang and J.-D. Zucker. Solving the multiple-instance problem: A lazy learning approach. In *ICML*, pp. 1119–1126, 2000.

X. Xu and E. Frank. Logistic regression and boosting for labeled bags of instances. In *PAKDD*, pp. 272–281, 2004.

S.-H. Yang, H. Zha, and B.-G. Hu. Dirichlet-bernoulli alignment: A generative model for multi-class multi-label multi-instance corpora. In *NIPS*, pp. 2143–2150, 2009.

Q. Zhang and S. A. Goldman. EM-DD: An improved multiple-instance learning technique. In *NIPS*, pp. 1073–1080, 2001.

Q. Zhang, S. A. Goldman, W. Yu, and J. E. Fritts. Content-based image retrieval using multiple-instance learning. In *ICML*, pp. 682–689, 2002.

Z.-H. Zhou and J.-M. Xu. On the relation between multi-instance learning and semi-supervised learning. In *ICML*, pp. 1167–1174, 2007.

Active Sensing as Bayes-Optimal Sequential Decision-Making

Sheeraz Ahmad
Computer Science and Engineering Dept.
University of California San Diego
La Jolla, CA 92093

Angela J. Yu
Cognitive Science Dept.
University of California San Diego
La Jolla, CA 92093

Abstract

Sensory inference under conditions of uncertainty is a major problem in both machine learning and computational neuroscience. An important but poorly understood aspect of sensory processing is the role of *active* sensing. Here, we present a Bayes-optimal inference and control framework for active sensing, C-DAC (Context-Dependent Active Controller). Unlike previously proposed algorithms that optimize abstract statistical objectives such as information maximization (Infomax) [Butko and Movellan, 2010] or one-step look-ahead accuracy [Najemnik and Geisler, 2005], our active sensing model directly minimizes a combination of *behavioral costs*, such as temporal delay, response error, and sensor repositioning cost. We simulate these algorithms on a simple visual search task to illustrate scenarios in which context-sensitivity is particularly beneficial and optimization with respect to generic statistical objectives particularly inadequate. Motivated by the geometric properties of the C-DAC policy, we present both parametric and non-parametric approximations, which retain context-sensitivity while significantly reducing computational complexity. These approximations enable us to investigate a more complex search problem involving peripheral vision, and we notice that the performance advantage of C-DAC over generic statistical policies is even more evident in this scenario.

1 Introduction

In the realm of symbolic problem solving, computers are sometimes comparable, or even better than, typical human performance. In contrast, in sensory processing, especially under conditions of noise, uncertainty, or non-stationarity, human performance is often still the gold standard [Martin et al., 2001, Branson et al., 2011]. One important tool the brain has at its disposal is *active sensing*, a goal-directed, context-sensitive control strategy that prioritizes sensing resources toward the most rewarding or informative aspects of the environment [Yarbus, 1967]. Most theoretical models of sensory processing presume *passiveness*, considering only how to represent or compute with given inputs, and not how to actively intervene in the input collection process itself, especially with respect to behavioral goals or environmental constraints. Having a formal understanding of active sensing is not only important for advancing neuroscientific progress but also for engineering applications, such as developing context-sensitive, interactive artificial agents.

The most well-studied aspect of human active sensing is saccadic eye movements, and early work suggests that saccades are attracted to *salient* targets that differ from surround in one or more of feature dimensions such as orientation, motion, luminance, and color contrast [Koch and Ullman, 1985, Itti and Koch, 2000]. This passive explanation does not take into account the fact that the observations made while attending the task can affect the fixations decisions that follow. More recently, there has been a shift to relax this constraint of passiveness, and the notion of saliency has been reframed probabilistically in terms of maximizing the informational gain (Infomax) given the spatial and temporal context [Lee and Yu, 2000, Itti and Baldi, 2006, Butko and Movellan, 2010]. Separately, in another active formulation, it has been proposed that saccades are chosen to maximize the greedy, one-step look-ahead probability of finding the target (greedy MAP), conditioned on self knowledge about visual acuity map [Najemnik and Geisler, 2005].

While both the Infomax and Greedy MAP algorithms brought a new level of sophistication – representing sensory processing as iterative Bayesian inference, quantifying the knowledge gain of different saccade choices, and incorporating knowledge about sensory noise – they are still limited in several key respects: (1) they optimize abstract computational quantities that do not directly relate to behavioral goals (eg, speed and accuracy) or task constraints (eg, cost of switching from one location to another); (2) relatedly, it is unclear how to adapt these algorithms to varying task goals (eg, locating someone in a crowd versus catching a moving object); (3) there is no explicit representation of time in these algorithms, and thus no means of trading off fixation duration or number of fixations with search accuracy. In the rest of the paper, we refer to Infomax and Greedy MAP as "statistical policies", in the sense that they optimize generic statistical objectives insensitive to behavioral objectives or contextual constraints.

In contrast to the statistical policies, we propose a Bayes-optimal inference and control framework for active sensing, which we call C-DAC (Context-Dependent Active Controller). Specifically, we assume that the observer aims to optimize a context-sensitive objective function that takes into account behavioral costs such as temporal delay, response error, and the cost of switching from one sensing location to another. C-DAC uses this objective to choose when and where to collect sensory data, based on a continually updated statistically optimal (Bayesian) representation of the sequentially collected sensory data. This framework allows us to derive behaviorally optimal procedures for making decisions about (1) where to acquire sensory inputs, (2) when to move from one observation location to another, and (3) how to negotiate the exploration-exploitation tradeoff between collecting additional data and terminating the observation process. We also compare the performance of C-DAC and the statistical policies under different task parameters, and illustrate scenarios in which the latter perform particularly poorly. Finally, we present two approximate value iteration algorithms, based on a low-dimensional parametric and non-parametric approximation of the value function, which retain context-sensitivity while significantly reducing computational complexity.

In Sec. 2, we describe in detail the C-DAC model. In Sec. 3, we apply the model to a visual search task, simulating scenarios where C-DAC achieves a flexible trade-off between speed, accuracy and effort depending on the task demands, whereas the statistical policies fall short – this forms experimentally testable predictions for future investigations. We also present approximate value-iteration algorithms, and an extension of the search problem that incorporates peripheral vision. We conclude with a discussion of the implications of this work, relationship to previous work, as well as pointers to future work (Sec. 4).

2 The Model: C-DAC

We consider a scenario in which the observer must produce a response based on sequentially observed noisy sensory inputs (e.g., identifying target location in a search task or scene category in a classification task), with the ability to choose *where* and *how long* to collect the sensory inputs.

2.1 Sensory Processing: Bayesian Inference

We use a Bayesian generative model to capture the observer's knowledge about the statistical relationship among hidden causes or variables and how they give rise to noisy sensory inputs, as well as prior beliefs of hidden variables. We assume that they use exact Bayesian inference in the *recognition model* to maintain a statistically optimal representation of the hidden state of the world based on the noisy data stream.

Conditioned on the target location (s, hidden) and the sequence of fixation locations ($\boldsymbol{\lambda}_t := \{\lambda_1, \ldots, \lambda_t\}$, known), the agent sequentially observes iid inputs ($\mathbf{x}_t := \{x_1, \ldots, x_t\}$):

$$p(\mathbf{x}_t|s;\boldsymbol{\lambda}_t) = \prod_{i=1}^{t} p(x_i|s;\lambda_i) = \prod_{i=1}^{t} f_{s,\lambda_i}(x_i) \qquad (1)$$

where $f_{s,\lambda}(x_t)$ is the likelihood function. These variables can be scalars or vectors, depending on the specific problem.

In the *recognition model*, repeated applications of Bayes' Rule can be used to compute the iterative posterior distribution over the k possible target locations, or the *belief state*:

$$\begin{aligned}
&\mathbf{p}_t := (P(s=1|\mathbf{x}_t;\boldsymbol{\lambda}_t), \ldots, P(s=k|\mathbf{x}_t;\boldsymbol{\lambda}_t)) \\
&\mathbf{p}_t^i = P(s=i|\mathbf{x}_t;\boldsymbol{\lambda}_t) \propto p(x_t|s=i;\lambda_t)P(s=i|\mathbf{x}_{t-1};\boldsymbol{\lambda}_{t-1}) \\
&\quad = f_{s,\lambda_t}(x_t)\mathbf{p}_{t-1}^i \qquad (2)
\end{aligned}$$

where $\mathbf{p}_0$ is the prior belief over target location.

2.2 Action Selection: Bayes Risk Minimization

The action selection component of active vision is a stochastic control problem where the agent chooses the sensing location and the number of data points collected, and we assume the agent can optimize this process dynamically based on ongoing data collection and size of sensory data, but the exact consequence of each action is not perfectly known ahead of time.

The goal is to find a good decision policy π, which maps the augmented belief state $(\mathbf{x}_t, \boldsymbol{\lambda}_t)$ into an action $a \in A$, where A consists of a set of termination actions, stopping and choosing a response, and a set of continuation actions, obtaining data point from a certain observation location. The policy π produces for each observation sequence $(x_1, \ldots, x_t, \ldots)$, a stopping time τ (number of data points observed), a sequence of fixation choices $\boldsymbol{\lambda}_\tau := (\lambda_1, \ldots, \lambda_\tau)$, and an eventual target choice δ.

In the Bayes risk minimization framework, the optimization problem is formulated in terms of minimizing an *expected* cost function, $L_\pi := \mathbb{E}[l(\tau, \boldsymbol{\lambda}_\tau, \delta)]_{\mathbf{x},s}$, averaged over stochasticity in the true target location s and the data samples $\mathbf{x}$. We assume that the cost incurred on each trial takes into account *temporal delay*, *switch cost (cost associated with each switch in sensing location)*, and *response error*, respectively. In accordance with the typical Bayes risk formulation of the sequential decision problem, we assume the cost function to be a linear combination of the relevant factors:

$$l(\tau, \delta; \boldsymbol{\lambda}_\tau, s) = c\tau + c_s n_\tau + \mathbf{1}_{\{\delta \neq s\}} \tag{3}$$

where n_τ is the total number of switches ($n_\tau := \sum_{t=1}^{\tau-1} \mathbf{1}_{\{\lambda_{t+1} \neq \lambda_t\}}$), c parameterizes the cost of temporal delay, c_s the cost of a switch, and unit cost for response errors is assumed (as we can always divide c and c_s by the appropriate constant to make it 1). The expected cost is $L_\pi := c\mathbb{E}[\tau] + c_s\mathbb{E}[n_s] + P(\delta \neq s)$, where the expectation is taken over τ, $\boldsymbol{\lambda}$, δ, and $\mathbf{x}_\tau$.

Bellman's dynamic programming equation [Bellman, 1952] tells us that the problem is optimized if at each time point, the agent chooses the action associated with the lowest expected cost (the *Q-factor* for that action), given his current knowledge or *belief state*, $\mathbf{p}_t$. The Q-factors for the stopping actions are straight forward: $\bar{Q}_t^i(\mathbf{p}_t, \boldsymbol{\lambda}_t) := \mathbb{E}[l(t, i)|\mathbf{p}_t, \boldsymbol{\lambda}_t] = ct + c_s n_t + (1 - \mathbf{p}_t^i)$. Obviously, the best stopping action δ is to minimize the probability of error. Thus, the stopping cost associated with the optimal stopping action ($i^* := \text{argmax}_i\ \mathbf{p}_t^i$) is:

$$\begin{aligned}\bar{Q}_t^*(\mathbf{p}_t, \boldsymbol{\lambda}_t) &:= \mathbb{E}[l(t, i^*)|\mathbf{p}_t, \boldsymbol{\lambda}_t] \\ &= ct + c_s n_t + (1 - \mathbf{p}_t^{i^*})\end{aligned} \tag{4}$$

The Q-factor associated with each continuation action j (continue sensing in location j) is:

$$\begin{aligned}Q_t^j(\mathbf{p}_t = \mathbf{p}, \boldsymbol{\lambda}_t) &:= c(t+1) + c_s(n_t + \mathbf{1}_{\{j \neq \lambda_t\}}) + \\ &\min_{\tau', \delta, \boldsymbol{\lambda}_{\tau'}} \mathbb{E}[l(\tau', \delta)|\mathbf{p}_0 = \mathbf{p}, \lambda_1 = j]\end{aligned} \tag{5}$$

with the optimal continuation action being $Q_t^* := \min_j Q_t^j = Q_t^{j^*}$. The expected cost of continuing observing in location j is equivalent to solving the original optimization problem with the prior belief set to the posterior after the previous t time-steps, and the first observation location being j. Suppose we define the *value function* $V(\mathbf{p}, i)$ as the expected cost associated with the optimal policy, given prior belief $\mathbf{p}_0 = \mathbf{p}$ and initial observation location $\lambda_1 = i$:

$$V(\mathbf{p}, i) := \min_{\tau, \delta, \boldsymbol{\lambda}_\tau} \mathbb{E}[l(\tau, \delta)|\mathbf{p}_0 = \mathbf{p}, \lambda_1 = i]\ . \tag{6}$$

Then the value function satisfies the following recursive relation:

$$\begin{aligned}V(\mathbf{p}, k) &= \min(\bar{Q}_1^*(\mathbf{p}, k), Q_1^*(\mathbf{p}, k)) \\ &= \min\Big(\Big(\min_i \bar{Q}_1^i(\mathbf{p}, k)\Big)\ , \\ &\quad \min_j \left(c + c_s\mathbf{1}_{\{j \neq k\}} + \mathbb{E}[V(\mathbf{p}', j)]\right)\Big)\end{aligned} \tag{7}$$

where $\mathbf{p}'$ is the belief state at next time-step, and the expectation is taken over the stochasticity in the next observation x. The optimal policy effectively divides the belief state space into a *stopping region* ($\bar{Q}^* \leq Q^*$) and a *continuation region* ($\bar{Q}^* > Q^*$), each of which further divided into subregions corresponding to alternative continuation and stopping actions. Note that the optimal decision policy is a *stationary* policy: the value function depends only on the belief state and observation location at the time the decision is to be taken, and not on time t *per se*.

Bellman's dynamic programming principle implies a numerical algorithm for computing the optimal policy: guess an initial setting $V'(\mathbf{p}, k)$ of the value function (e.g., minimal stopping cost associated with each belief state $\mathbf{p}$ and observation location k), then iterate Eq. 7 until convergence, which yields the value function $V(\mathbf{p}, k) = V^\infty(\mathbf{p}, k)$.

3 Case Study: Visual Search

In this section, we apply the active sensing model to a simple, three location visual search task, where we can compute the exact optimal policy (up to discretization of the state space), and compare its performance with the statistical policies [Butko and Movellan, 2010, Najemnik and Geisler, 2005]. The target and distractors differ in terms of the likelihood of observations received, when looking at them.

3.1 C-DAC Policy

For simplicity, we assume that the observations are binary and Bernoulli distributed (iid conditioned on target and fixation locations):

$$p(x|s = i; \lambda_t = j) = \mathbf{1}_{\{i=j\}}\beta_1^x(1-\beta_1)^{1-x} + \mathbf{1}_{\{i \neq j\}}\beta_0^x(1-\beta_0)^{1-x}$$

The difficulty of the task is determined by the discriminability between target and distractor, or the difference between β_1 and β_0. For simplicity, we assume that the only stopping action available is to choose the current fixated location: $\hat{s}(\tau; \lambda_\tau = j) = j$. To reduce the parameter space, we also set $\beta_0 = 1 - \beta_1$, which is a reasonable assumption stating that the distractor and target stimuli only differ in one way (e.g. opposing direction of motion when using random dots stimulus with the coherence of dots kept the same). In the following, we first present a brief description of the greedy MAP and the infomax algorithms, before moving on to model comparisons.

3.2 Greedy MAP Policy

The *greedy MAP* algorithm [Najemnik and Geisler, 2005] suggests that agents should try to maximize the expected one-step look-ahead probability of finding the target. Thus, the reward function is:

$$\begin{aligned} R^g(\mathbf{p}_t, j) &= \mathbb{E}_{x_{t+1}}[\max_i P(s = i|\mathbf{x}_t, x_{t+1}, \boldsymbol{\lambda}_t, \lambda_{t+1} = j)] \\ &= \mathbb{E}_{x_{t+1}}[\max_i (\mathbf{p}^i_{t+1})|x_{t+1}, \lambda_{t+1} = j] \end{aligned}$$

To keep the notations consistent, we define the associated *Q-factor*, cost and policy as:

$$\begin{aligned} Q^g(\mathbf{p}_t, j) &= -R^g(\mathbf{p}_t, j) \\ V^g(\mathbf{p}_t, j) &= \min_j Q^g(\mathbf{p}_t, j) \\ \lambda^g_{t+1} &= \operatorname*{argmin}_j Q^g(\mathbf{p}_t, j) \end{aligned}$$

3.3 Infomax Policy

The *infomax* algorithm [Butko and Movellan, 2010] tries to maximize the information gained from each fixation, by minimizing the expected cumulative future entropy. Similar to [Butko and Movellan, 2010], we can define the *Q-factors*, cost and the policy as:

$$\begin{aligned} Q^{im}(\mathbf{p}_t, j) &= \sum_{t'=t+1}^{T} \mathbb{E}_{x_{t'}}[H(\mathbf{p}_{t'})|x_{t'}, \lambda_{t+1} = j] \\ V^{im}(\mathbf{p}_t, j) &= \min_j Q^{im}(\mathbf{p}_t, j) \\ \lambda^{im}_{t+1} &= \operatorname*{argmin}_j Q^{im}(\mathbf{p}_t, j) \end{aligned}$$

where $H(\mathbf{p}) = -\sum_i \mathbf{p}^i \mathbf{log} \mathbf{p}^i$ is Shannon's entropy. Note that neither the original greedy MAP nor the infomax algorithm provide a principled answer as to when to stop searching and respond. They need to be augmented to stop once the maximum probability of any location containing the target exceeds a fixed threshold. We come back to the problem of how we set this threshold when we present comparison results.

3.4 Model Comparison

Before we discuss the performance of different models in terms of "behavioral" output, we first visually illustrate the decision policies (Fig. 1). The belief state $\mathbf{p}$ is represented by discretizing the two-dimensional belief state space $(\mathbf{p}^1, \mathbf{p}^2)$ with $m = 201$ bins in each dimension ($\mathbf{p}^3 = 1 - \mathbf{p}^1 - \mathbf{p}^2$). Although for C-DAC the policy also depends on the current fixation location, we only show it for fixating the first location; the other representations being rotationally symmetric. In Fig. 1, the parameters used for the C-DAC policy are $(c, c_s, \beta) = (0.1, 0, 0.9)$, and for the statistical policies, $(\beta, \mathit{thresh}) = (0.9, 0.8)$. Note that for this simple scenario with no switch cost, the infomax policy looks almost like the C-DAC policy – fixate the most likely location unless there is very strong evidence that the fixated location contains the target, in which case the observer should stop. The greedy MAP policy, on the other hand, looks completely different, and is in fact *ambiguous* in the sense that for a large set of belief states the policy does not give a unique next fixation location. We show one instance of this seemingly random policy, and note that there are regions where the policy suggests to look at either location 1 or 2 or 3 (corner regions speckled with green, orange and brown). Similarly, there are regions where the policy suggests to look at 1 or 2 (green+orange region). In fact, the performance of greedy MAP is so poor that we exclude it from the model comparisons below.

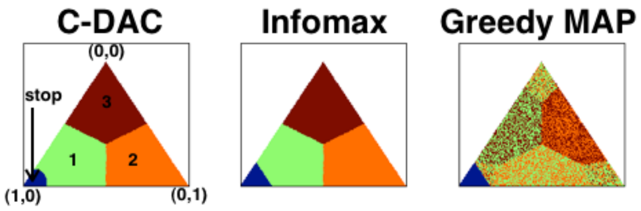


Figure 1: Decision policies – Infomax resembles C-DAC. Blue: stop. Green: fixate location 1. Orange: fixate location 2. Brown: fixate location 3. Environment $(c, c_s, \beta) = (0.1, 0, 0.9)$. Threshold for infomax and greedy MAP = 0.8

Fig. 2 shows the effects of how the C-DAC policy changes when different parameters of the task are changed. As seen in the figure, the stopping region expands if the cost of time increases (high c), intuitively this makes sense – if each time step is costlier then the observer should stop at a lower level of confidence, at the expense of higher error rate. Similarly, for the case when β is smaller (high noise), stopping with a lower level of confidence makes sense – the value of each additional observation depends on how noisy the data is, the noisier the less worthwhile to continue observing, thus leading to a lower stopping criterion. Lastly, and arguably the most interesting case, is when there is an additional switch cost (added c_s); this deters the algorithm from switching even when the belief in a given location has reduced below 1/3. In fact, this is the scenario where optimizing for behavioral objectives turns out to be truly beneficial, and although infomax can approximate the C-DAC policy when the switch cost is 0, it cannot do so when switch cost comes in to play.

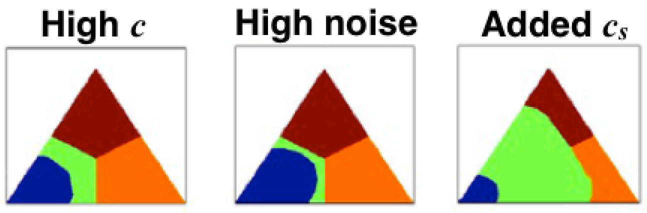


Figure 2: C-DAC policy for different environments (c, c_s, β) – high c $(0.2, 0, 0.9)$, high noise $(0.1, 0, 0.7)$, and added c_s $(0.1, 0.1, 0.9)$.

Next, we look at how these intuitions from the policy plots translate to output measures in terms of accuracy, response delay, and number of fixations. In order to set the stopping threshold for the infomax policy in the most generous/optimistic setting, we first run the C-DAC policy, and then set the threshold for infomax so that it matches the accuracy of C-DAC [1], while we compare the other output measures. We choose two scenarios: (1) no switch cost, (2) with switch cost. For all simulations, the algorithm starts with uniform prior $(\mathbf{p} = (1/3, 1/3, 1/3))$ and initial fixation location 1, while the true target location is uniformly distributed. Fig. 3 shows the accuracy, number of time steps and number of switches for both scenarios. Confirming the intuition from the policy plots, the performance of infomax and C-DAC are comparable for $c_s = 0$. However, when a switch cost is added, $c_s = 0.2$, we see that although the accuracy is comparable by design, there is small improvement in search time of C-DAC, and a notable advantage in the number of switches. The behavior of the infomax policy does not adapt to the change in the behavioral cost function, thus incurring an overall higher cost. Algorithms like infomax that maximize abstract statistical objectives lack the inherent flexibility to adapt to changing behavioral goals or environmental constraints. Even for this simple visual search example, Infomax does not have a principled way of setting the stopping threshold, and we gave it the best-scenario outcome by adopting the stopping policy generated by C-DAC in different contexts.

[1]Since a binary search is required to set this matching threshold, and the accuracy is sensitive w.r.t. this threshold, we settle on an approximate accuracy match for infomax that is comparable or lower than C-DAC.

3.5 Approximate Control

Our model is formally a variant of POMDP (Partially Observable Markov Decision Process), or, more specifically, a Mixed Observability Markov Decision Process (MOMDP) [Ong et al., 2010, Araya-López et al., 2010], which differs from ordinary POMDP in that part of the state space is partly hidden (target location in our case) and partly observable (current fixation location in our case). In general, POMDPs are hard to solve since the decision made at each time step depends on all the past actions and observations, thus imposing enormous memory requirements. This is known as the curse of history, and is the first major hurdle towards any practical solution. An elegant way to alleviate this is to use belief states which serve as a sufficient statistic for the process history, thus requiring to maintain just a single distribution instead of the entire history. Converting a POMDP to a belief-state MDP is in fact a prevalent technique and the one we employ. However, this leads to another computational hurdle, known as the curse of dimensionality, since now we have a MDP with a continuous state-space, making tabular representation of value function infeasible. One way to work around the problem is to discretize the belief state space into a grid, where instead of finding the value function at all the points in the belief state simplex, we only do so for a finite number of grid points. The grid approximation, that we also use, has appealing performance guarantees which improve as the density of the grid is increased [Lovejoy, 1991]. To evaluate the value function at the points not in this set, we use some sort of interpolation technique (value at the nearest grid point, weighted average value at k-nearest grid point, etc.). However, although grid approximation may work for small state spaces, it does not scale well to larger, practical problems. For example, when used for the active sensing problem with k sensing locations, a uniform grid of size n has $O(kn^{k-1})$ complexity.

Although there is a rich body of literature on approximate solutions of POMDP (e.g. [Powell, 2007, Lagoudakis and Parr, 2003, Kaplow, 2010]) tackling both general as well as application-specific approximations, most are inappropriate for dealing with the MOMDP problem such as the one encountered here. Furthermore, most of the POMDP approximation algorithms focus on discounted rewards and/or finite-horizon problems. Our formulation does not fall into these categories and thus require novel approximation schemes. We note that the Q-factors and the resulting value function are smooth and concave, making them amenable to low dimensional approximations. At each step, we find a low dimensional representation of the value function, and use that for the update step of the

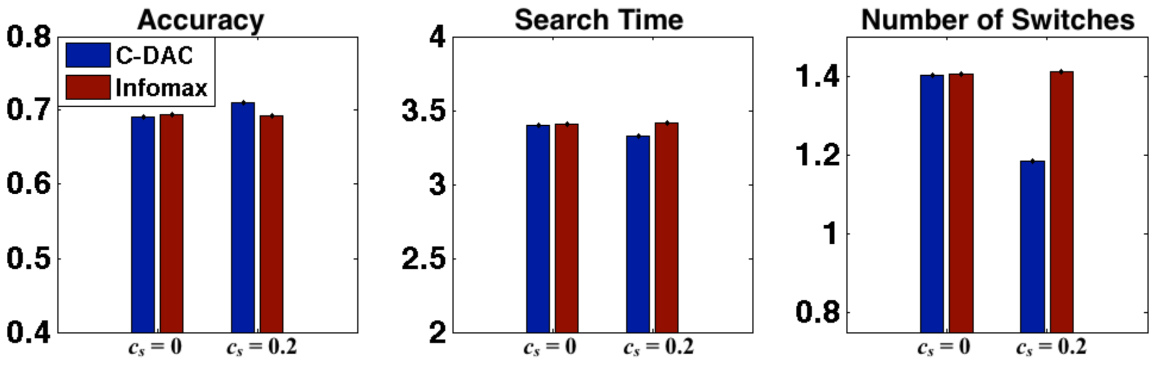


Figure 3: Comparison between C-DAC and Infomax for two environments $(c, c_s, \beta) = (0.1, 0, 0.8)$ and $(0.1, 0.2, 0.8)$. C-DAC has superior performance when $c_s > 0$.

value iteration algorithm. Specifically, instead of recomputing the value function at each grid point, here we generate a large number of samples uniformly on the belief state space, compute a new estimate of the value function at those locations, and then extrapolate the value function to everywhere by improving its parametric fit.

The first low-dimensional approximation we consider is the Radial Basis Functions (RBF) representation:

1. Generate M RBFs, centered at $\{\mu_i\}_{i=1}^{M}$, with fixed σ: $\phi(\mathbf{p}) = \frac{1}{\sigma(2\pi)^{k/2}} e^{\frac{||\mathbf{p}-\mu_i||^2}{2\sigma^2}}$
2. Generate m random points from belief space, $\mathbf{p}$.
3. Initialize $\{V(\mathbf{p}_i)\}_{i=1}^{m}$ with the stopping costs.
4. Find minimum-norm $\mathbf{w}$ from: $V(\mathbf{p}) = \Phi(\mathbf{p})\mathbf{w}$.
5. Generate new m random belief state points $(\mathbf{p}')$.
6. Evaluate required V values using current $\mathbf{w}$.
7. Update $V(\mathbf{p}')$ using value iteration.
8. Find a new $\mathbf{w}$ from $V(\mathbf{p}') = \Phi(\mathbf{p}')\mathbf{w}$.
9. Repeat steps 5 through 8, until w converges.

While we adopt a Gaussian kernel function, other constructs are possible and have been implemented in our problem without significant performance deviation (not shown), e.g. multiquadratic ($\phi(\mathbf{p}) = \sqrt{1 + \epsilon||\mathbf{p} - \mu_i||^2}$), inverse-quadratic($\phi(\mathbf{p}) = (1 + \epsilon||\mathbf{p} - \mu_i||^2)^{-1}$), thin plate spine ($\phi(\mathbf{p}) = ||\mathbf{p} - \mu_i||^2 \ln||\mathbf{p} - \mu_i||$), etc. [Buhmann, 2003].

The RBF approximation requires setting several parameters (number, mean, and variance of bases), which can be impractical for large problems, when there is little or no information available about the properties of the true value function. We thus also implement a nonparametric variation of the algorithm, whereby we use Gaussian Process Regression (GPR) [Williams and Rasmussen, 1996] to estimate the value function (step 4, 6 and 8). In addition, we also implement GPR with hyperparameter learning (Automatic Relevance Determination, ARD), thus obviating the need to preset model parameters.

The approximations lead to considerable computational savings. The complexity of the RBF approximation is $O(k(mM + M^3))$, for k sensing locations, m random points chosen at each step, and M bases. For the GPR approximation, the complexity is $O(kN^3)$, where N is the number of points used for regression. In practice, all the approximation algorithms we consider converge rapidly (under 10 iterations), though we do not have a proof that this holds for a general case.

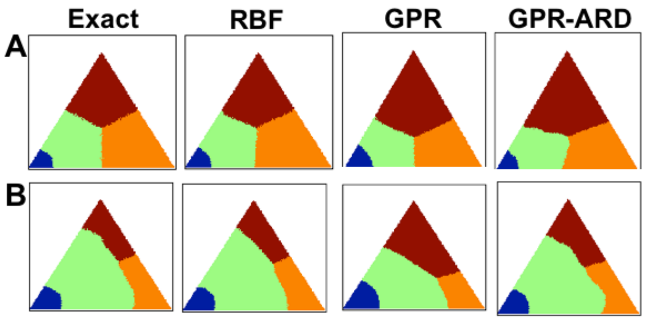


Figure 4: Exact vs. approximate policies shown over $n = 201$ bins. (A) Environment $(c, c_s, \beta) = (0.1, 0, 0.9)$. (B) $(c, c_s, \beta) = (0.1, 0.1, 0.9)$.

In the simulations, the RBF approximate policy uses $m = 1000$ random point for each iteration, and $M = 49$ bases, uniformly placed in the belief simplex, with a unit variance. The GPR approximate policy uses a unit length scale, unit signal strength and a noise-strength of 0.1, with $N = 200$ random points used for regression. Fig. 4A shows the exact policy vs. the learned approximate policies for different approximations when the switch cost is 0, $(c, c_s, \beta) = (0.1, 0, 0.9)$. We notice that with handcrafted bases, RBF is a good approximation of the exact policy, whereas relaxing the parametric form in GPR and subsequently learning the hyperparameters in GPR with ARD, leads to a slightly poorer but more robust non-parametric approximation. Similar observations can be made in Fig. 4B, for the environment with added switch cost, $(c, c_s, \beta) = (0.1, 0.1, 0.9)$. All the results are shown over a 201x201 grid. These faster yet robust approximations motivated us to apply our model to more complex problems. We investigate one such problem of visual search with peripheral vision next, and show how our model is fundamentally different from existing formulations such as infomax, even when the cost of effort is not considered.

3.6 Visual Search with Peripheral Vision

In the very simple three-location visual search problem we considered above, we did not incorporate the possibility of peripheral vision, or the more general possibility that a sensor positioned in a particular location can have distance-dependent, degraded information about nearby locations as well. We therefore consider a simple example with peripheral vision (see Fig. 5B), whereby the observer can saccade to intermediate locations that give reduced information about either two (sensing locations on the edges of the triangle) or three (sensing location in the center) stimuli. This is motivated by experimental observations that humans not only fixate most probable target locations but sometimes also center-of-gravity locations that are intermediate among two or more target locations [Findley, 1982, Zelinsky et al., 1997].

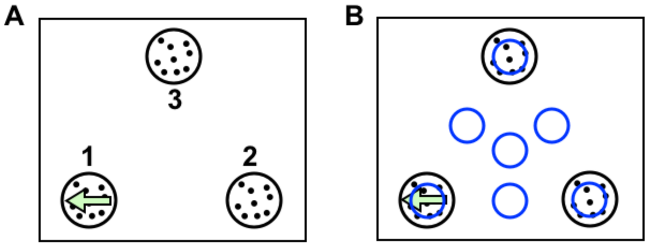


Figure 5: Schematics of visual search task. The general task is to find the target (left-moving dots) amongst distractors (right-moving dots). Not drawn to scale. (A) Task 1: agent fixates one of the target patches at any given time. (B) Task 2: agent fixates one of the blue circle regions at any given time

Formally, we need an *acuity map*, the notion that it is possible to gain information about stimuli peripheral to the fixation center (fovea), such that the quality of that information decays at greater spatial distance away from the fovea. For example, the task of Fig. 5B would require a continuation action space of 7 elements, $L = \{l_1, l_2, l_3, l_{12}, l_{23}, l_{13}, l_{123}\}$, where the first three actions correspond to fixating one of the three target locations, the next three to fixating midway between two target locations, and the last to fixating the center of all three. We parameterize the quality of peripheral vision by augmenting the observations to be three-dimensional, (x^1, x^2, x^3), corresponding to the three simultaneously viewed locations. We assume that each x_i is generated by a Bernoulli distribution favoring 1 if it is the target, and 0 if it is not, and its magnitude (absolute difference from 0.5) is greatest when observer directly fixates the stimulus, and smallest when the observer directly fixates one of the other stimuli. We use 4 parameters to characterize the observations ($1 > \beta_1 > \beta_2 > \beta_3 > \beta_4 >= 0.5$). So, when the agent is fixating one of the potential target locations (l_1, l_2 or l_3), it gets an observation from the fixated location (parameter β_1 or $1 - \beta_1$ depending on whether it is the target or a distractor), and observations from the non-fixated locations (parameter β_4 or $1 - \beta_4$ depending on whether they are a target or a distractor). Similarly, for the midway locations (l_{12}, l_{23} or l_{13}), the observations are received for the closest locations (parameter β_2 or $1 - \beta_2$ depending on whether they are a target or a distractor), and from the farther off location (parameter β_4 or $1 - \beta_4$ depending on whether it is the target or a distractor). Lastly, for the center location (l_{123}), the observations are made for all three locations (parameter β_3 or $1 - \beta_3$ depending on whether they are a target or a distractor). Furthermore, since the agent can now look at locations that cannot be target, we relax the assumption that the agent must look at a particular location before choosing it, allowing the agent to stop at any location and declare the target.

3.7 Model Comparison

We first present the policies, and, similar to our discussion of simple visual search task, we only show the C-DAC policy looking at the first location (l_1) (the other fixation-dependent policies are rotationally symmetric). It is evident from Fig. 6 that now the C-DAC policy differs from the infomax policy even when no switch cost is considered, thus pointing to a more fundamental difference between the two. Note that for the parameters used here, C-DAC never chooses to look at the center l_{123}, but it does so for other parameter settings (not shown). Infomax, however, never even looks at the actual potential locations, favoring only midway locations before declaring the target location.

For performance comparison in terms of behavioral output, we again investigate two scenarios: (1) no switch cost, (2) with switch cost. The threshold for infomax is set so that the accuracies are matched to facilitate fair comparison. For all simulations, the al-

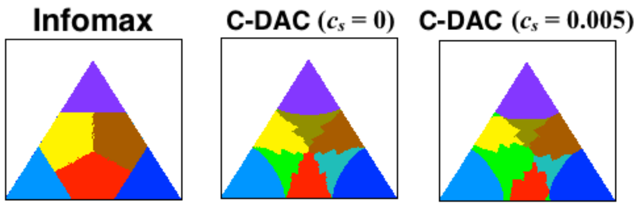


Figure 6: Decision policies. Azure: stop and choose location l_1. Blue: stop and choose l_2. Indigo: stop and choose l_3. Green: fixate l_1. Sea-green: fixate l_2. Olive: fixate l_3. Red: fixate l_{12}. Brown: fixate l_{23}. Yellow: fixate l_{13}. Environment $(c, \beta 1, \beta_2, \beta_3, \beta_4) = (0.05, 0.62, 0.6, 0.55, 0.5)$. Threshold for infomax $= 0.6$

gorithm starts with uniform prior ($\mathbf{p} = (1/3, 1/3, 1/3)$) and initial fixation at the center (location l_{123}), while the true target location is uniformly distributed. Fig. 7 shows the accuracy, number of time steps, and number of switches for both scenarios. Now we notice that C-DAC outperforms infomax even when switch cost is not considered, in contrast to the simple task without peripheral vision (Fig. 3). Note however that C-DAC makes more switches for $c_s = 0$, which makes sense since switches have no cost, and search time can potentially be reduced by allowing more switches. However, when we add a switch cost ($c_s = 0.005$), C-DAC significantly reduces number of switches, whereas infomax lacks this adaptability to a changed environment.

4 Discussion

In this paper, we proposed a POMDP plus Bayes risk-minimization framework for active sensing, which optimizes behaviorally relevant objectives in expectation, such as speed, accuracy, and switching efficiency. We compared this C-DAC policy to the previously proposed *infomax* and *greedy MAP* policies. We found that greedy MAP performs very poorly, and although Infomax can approximate the optimal policy for some simple environments, it lacks intrinsic context sensitivity or flexibility. Specifically, for different environments, there is no principled way to set a decision threshold for either greedy MAP or Infomax, leading to higher costs, longer fixation durations, and larger number of switches in problem settings when those costs are significant. This performance difference and the advantage of the added flexibility provided by C-DAC becomes even more profound when we consider a more general visual search problem with peripheral vision. The family of approximations that we present opens up the avenue for application of our model to complex, real world problems.

There have been several other related active sensing algorithms that differ from C-DAC in their state representation, inference, control and/or approximation scheme. We briefly summarize some of these here. In [Darrell and Pentland, 1996], the problem of active gesture recognition is studied, by using historic state representation and nearest neighbor Q-function approximation. Sensing strategies for robots in RoboCup competition is studied in [Kwok and Fox, 2004], which uses states augmented with associated uncertainty and model-free Least Square Policy Iteration (LSPI) approximation [Lagoudakis and Parr, 2003]. Context dependent goals are considered in [Ji et al., 2007] and [Naghshvar and Javidi, 2010]. The former concentrates on multi-sensor multi-aspect sensing using Point Based Value Iteration (PBVI) approximation [Pineau et al., 2006]. The latter aims to provide conditions for reduction of an active sequential hypothesis testing problem to passive hypothesis testing. A Reinforcement Learning paradigm where reward is not dependent on information gain but on how close a saccade brings the target to the optical axis has also been proposed [Minut and Mahadevan, 2001]. Other control strategies like random search, sequential sweeping search, "Drosophila-inspired" search [Chung and Burdick, April 2007] and hierarchical POMDPs for visual action planning [Sridharan et al., 2010] have also been proposed. We choose infomax to compare our C-DAC policy against because, as a human-vision inspired model, it not only explains human fixation behavior on a variety of tasks, but also has cutting edge computer vision applications (e.g. the digital eye [Butko and Movellan, 2010]).

A related problem domain, not typically studied as POMDP or MDP, is Multi-Armed Bandits (MAB) [Gittins, 1979]. The classical example of a MAB problem concerns with pulling levers (or playing arms) in a set of slot machines. The person gambling is unaware of the states and reward distribution of the levers, and has to figure out which lever to pull next in order to maximize the cumulative reward. Noting a correspondence between the ideas of pulling arms and fixating location, and between rewards and observations, the MAB framework seems to describe the active sensing problem. Concretely, given the locations fixated (arms played) so far, and the observations (rewards) received, how to choose which location to fixate (which arm to play) next. However, there are certain characteristics of the active sensing problem that make it difficult to study in a MAB framework as yet. Firstly, the problem is an instance of restless bandits [Whittle, 1988], where the state of an arm can change even when it is not played. In active sensing, the belief about a location being the target does change even when it is not fixated. Whittles index is a simple rule that assigns a value to each arm in a restless setting, and the

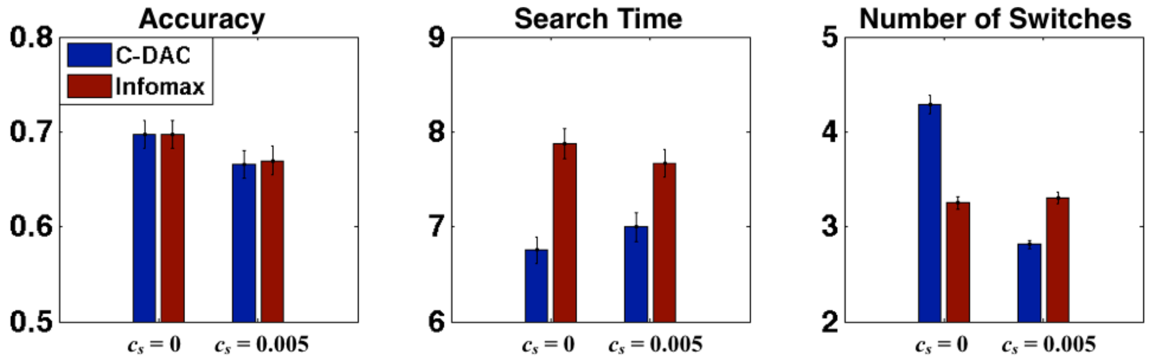


Figure 7: Comparison between C-DAC and Infomax on Task 2 for two environments $(c, \beta 1, \beta_2, \beta_3, \beta_4) = (0.05, 0.62, 0.6, 0.55, 0.5)$, $c_s = 0$ and $c_s = 0.005$. C-DAC adjusts the time steps and number of switches depending on the environment, taking a little longer but reducing number of switches when effort has cost.

arm with the highest value is then played. The rule is asymptotically optimal only for a sub-class of problems (e.g. [Washburn and Schneider, 2008] and [Liu and Zhao, 2010]), but not optimal in general. Secondly, the states of the arms in the active sensing task are correlated (the elements of the belief-state have to add up to 1). There is some work on correlated arms for specific structure of correlation, like clustered arms [Pandey et al., 2007] and Gaussian process bandits [Dorard et al., 2009], but so far there is no general strategy for handling this scenario.

Active learning is another related approach, with hypothesis testing as a sub-problem that is related to the problem of active sensing. The setting involves an unknown true hypothesis, and an agent that can perform queries providing information about the underlying hypothesis. The task is then to determine which query to perform next to optimally reduce the number of plausible hypothesis (version space). In active sensing however, although the belief about a hypothesis (target location) can become arbitrarily low, the number of plausible hypothesis does not reduce. This problem is investigated in [Golovin et al., 2010], and a near-optimal greedy solution is proposed along with performance guarantees. Besides the sub-optimality of the approach, the same test cannot be performed more than once (whereas in active sensing, one location can be fixated more than once). The lack of this provision stems from the fact that the noisy observations considered are actually deterministic with respect to a hidden noise parameter. Thus, as of yet it is hard to cast the active sensing problem in this framework.

We thus conclude that although there is a rich body of literature on related problems, as can be seen from the few examples we presented, our formulation is novel (to our best knowledge) in its goals and principled approach to the problem of active sensing. In general, the framework proposed here has the potential for not only applications in visual search, but a host of other problems, ranging from active scene categorization to active foraging. The decision policies it generates are adaptive to the environment and sensitive to contextual factors. This flexibility and robustness to different environments makes the framework an appealing choice for a variety of active sensing applications.

References

Mauricio Araya-López, Vincent Thomas, Olivier Buffet, and François Charpillet. A closer look at momdps. In *Tools with Artificial Intelligence (ICTAI), 2010 22nd IEEE International Conference on*, volume 2, pages 197–204. IEEE, 2010.

R Bellman. On the theory of dynamic programming. *PNAS*, 38(8):716–719, 1952.

S Branson, P Perona, and S Belongie. Strong supervision from weak annotation: Interactive training of deformable part models. *IEEE International Conference on Computer Vision*, 2:1832–1839, 2011.

M.D. Buhmann. *Radial basis functions: theory and implementations*, volume 12. Cambridge university press, 2003.

N J Butko and J R Movellan. Infomax control of eyemovements. *IEEE Transactions on Autonomous Mental Development*, 2(2):91–107, 2010.

Timothy H. Chung and Joel W. Burdick. A decision-making framework for control strategies in probabilistic search. *Intl. Conference on Robotics and Automation. ICRA*, April 2007.

T. Darrell and A. Pentland. Active gesture recognition using partially observable markov decision processes. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 3, pages 984–988. IEEE, 1996.

L. Dorard, D. Glowacka, and J. Shawe-Taylor. Gaussian process modelling of dependencies in multi-armed bandit problems. In *Int. Symp. Op. Res*, 2009.

J M Findley. Global processing for saccadic eye movements. *Vision Research*, 1982.

J C Gittins. Bandit processes and dynamic allocation indices. *J. Royal Stat. Soc.*, 41:148–77, 1979.

D. Golovin, A. Krause, and D. Ray. Near-optimal bayesian active learning with noisy observations. *arXiv preprint arXiv:1010.3091*, 2010.

L Itti and P Baldi. Bayesian surprise attracts human attention. In *Advances in Neural Information Processing Systems, Vol. 19*, pages 1–8, Cambridge, MA, 2006. MIT Press.

L Itti and C Koch. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision Research*, 40(10-12):1489–506, 2000.

S. Ji, R. Parr, and L. Carin. Nonmyopic multiaspect sensing with partially observable markov decision processes. *Signal Processing, IEEE Transactions on*, 55(6):2720–2730, 2007.

R. Kaplow. *Point-based POMDP solvers: Survey and comparative analysis*. PhD thesis, McGill University, 2010.

C Koch and S Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. *Hum. Neurobiol.*, 1985.

C. Kwok and D. Fox. Reinforcement learning for sensing strategies. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3158–3163. IEEE, 2004.

M.G. Lagoudakis and R. Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

Tai Sing Lee and Stella Yu. An information-theoretic framework for understanding saccadic behaviors. In *Advance in Neural Information Processing Systems*, volume 12, Cambridge, MA, 2000. MIT Press.

K. Liu and Q. Zhao. Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access. *Information Theory, IEEE Transactions on*, 56(11):5547–5567, 2010.

W.S. Lovejoy. Computationally feasible bounds for partially observed markov decision processes. *Operations research*, 39(1):162–175, 1991.

D Martin, C Fowlkes, D Tal, and J Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th International Conference on Computer Vision*, volume 2, pages 416–423, 2001.

S Minut and S Mahadevan. A reinforcement learning model of selective visual attention. In *Proceedings of the Fifth International Conference on Autonomous Agents, Montreal*, 2001.

M. Naghshvar and T. Javidi. Active m-ary sequential hypothesis testing. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 1623–1627. IEEE, 2010.

J Najemnik and W S Geisler. Optimal eye movement strategies in visual search. *Nature*, 434(7031):387–91, 2005.

Sylvie CW Ong, Shao Wei Png, David Hsu, and Wee Sun Lee. Planning under uncertainty for robotic tasks with mixed observability. *The International Journal of Robotics Research*, 29(8):1053–1068, 2010.

S. Pandey, D. Chakrabarti, and D. Agarwal. Multi-armed bandit problems with dependent arms. In *Proceedings of the 24th international conference on Machine learning*, pages 721–728. ACM, 2007.

J. Pineau, G. Gordon, and S. Thrun. Anytime point-based approximations for large pomdps. *Journal of Artificial Intelligence Research*, 27(1):335–380, 2006.

W.B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. Wiley-Interscience, 2007.

M Sridharan, J Wyatt, and R Dearden. Planning to see: A hierarchical approach to planning visual actions on a robot using pomdps. *Artificial Intelligence*, pages 704–725, 2010.

R. Washburn and M. Schneider. Optimal policies for a class of restless multiarmed bandit scheduling problems with applications to sensor management. *Journal of Advances in Information Fusion. v3*, 2008.

P Whittle. Restless bandits: activity allocation in a changing world. *J. App. Probability*, 25A:287–98, 1988.

C K I Williams and C E Rasmussen. Gaussian processes for regression. In M.C. Mozer D. S. Touretzky and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 514–20. MIT Press, Cambridge, MA, 1996.

A F Yarbus. *Eye Movements and Vision.* Plenum Press, New York, 1967.

G J Zelinsky, R P Rao, M M Hayhoe, and D H Ballard. Eye movements reveal the spatio-temporal dynamics of visual search. *Psychol. Sci.*, 1997.

The Bregman Variational Dual-Tree Framework

Saeed Amizadeh
Intelligent Systems Program
University of Pittsburgh
Pittsburgh, PA 15213

Bo Thiesson
Department of Computer Science
Aalborg University
Aalborg, Denmark

Milos Hauskrecht
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15213

Abstract

Graph-based methods provide a powerful tool set for many non-parametric frameworks in Machine Learning. In general, the memory and computational complexity of these methods is quadratic in the number of examples in the data which makes them quickly infeasible for moderate to large scale datasets. A significant effort to find more efficient solutions to the problem has been made in the literature. One of the state-of-the-art methods that has been recently introduced is the Variational Dual-Tree (VDT) framework. Despite some of its unique features, VDT is currently restricted only to Euclidean spaces where the Euclidean distance quantifies the similarity. In this paper, we extend the VDT framework beyond the Euclidean distance to more general Bregman divergences that include the Euclidean distance as a special case. By exploiting the properties of the general Bregman divergence, we show how the new framework can maintain all the pivotal features of the VDT framework and yet significantly improve its performance in non-Euclidean domains. We apply the proposed framework to different text categorization problems and demonstrate its benefits over the original VDT.

1 Introduction

Graph-based methods provide a powerful tool set for many non-parametric frameworks in Machine Learning (ML). The common assumption behind these methods is the datapoints can be represented as the nodes of a graph whose edges encode some notion of *similarity* between the datapoints. Graph-based methods have been applied to various applications in ML including clustering (Ng et al., 2001a; von Luxburg, 2007; Amizadeh et al., 2012b), semi-supervised learning (Zhu, 2005; Zhou et al., 2003), link-analysis (Ng et al., 2001c,b) and dimensionality reduction (Belkin and Niyogi, 2002; Zhang et al., 2012).

On the algorithmic side, many of these methods involve computing the *random walk* on the graph which is mathematically represented by a *(Markov) transition matrix* over the graph. In general, the computation of such matrix takes $O(N^2)$ time and memory, where N is the problem size. As a result, for problems with large N, the direct computation of the transition matrix (or its variants) quickly becomes infeasible. This is indeed a big challenge for applying many graph-based frameworks specially with the advent of large-scale datasets in many fields in ML. To tackle this challenge, a significant effort has been made in the literature to develop the approximation techniques that somehow *reduce* the representation of the underlying graph. Based on the nature of approximation, these methods are generally categorized into node reduction (Kumar et al., 2009; Talwalkar et al., 2008; Amizadeh et al., 2011), edge reduction (von Luxburg, 2007; Jebara et al., 2009; Qiao et al., 2010), Fast Gauss Transform (Yang et al., 2003, 2005) and hierarchical (Amizadeh et al., 2012a; Lee et al., 2011) techniques.

Recently, Amizadeh et. al (Amizadeh et al., 2012a) have proposed a hierarchical approximation framework called the Variational Dual-Tree (VDT) framework for the same purpose. In particular, by combining the variational approximation with hierarchical clustering of data, VDT provides a fast and scalable method to *directly* approximate the transition matrix of the random walk. Furthermore, the hierarchical nature of VDT makes it possible to have approximations at different levels of granularity. In fact, by changing the level of approximation in VDT, one can adjust the trade-off between computational complexity and approximation accuracy. One major restriction with the VDT framework, however, is it only works for Euclidean spaces where similarity is expressed via the Euclidean distance. This can be problematic in many real applications where the Euclidean distance is not the best way to encode similarity. Unfortunately, the extension of VDT to a general distance metric is *not* straightforward, mainly because the key computational gain of VDT is achieved by relying on the functional form of the Euclidean distance.

In this paper, our goal is to extend the VDT framework to use a general class of divergences called *Bregman divergences* (Banerjee et al., 2005). We propose this new method as the *Bregman Variational Dual Tree (BVDT)* framework. Bregman divergences cover a diverse set of divergences and distances which can all be reformulated in a unified functional form. They have been used in many ML paradigms including clustering (Banerjee et al., 2005), matrix approximation (Dhillon and Sra, 2005; Banerjee et al., 2004), nearest neighbor retrieval (Cayton, 2008) and search (Zhang et al., 2009). From the applied side, some famous distances and divergences such as Euclidean distance, KL-Divergence and Logistic Loss are in fact the instances of Bregman divergences. By extending the VDT framework to Bregman divergences, one can use it with any instance of Bregman divergences depending on the application and therefore it becomes accessible to a large class of applications where similarity is expressed via non-Euclidean measures.

The crucial aspect of using Bregman divergences for the VDT framework is it does not cost us any extra order of computations; it still has the same computational and memory complexity order as the original VDT. In particular, we show that by exploiting the functional form of the general Bregman divergence, one can design a similar mechanism as in VDT to significantly cut unnecessary distance computations. This is a very important property because the motivation to develop variational dual-trees in the first place was to tackle large-scale problems.

One nice feature of the VDT framework is its probabilistic interpretation. In fact, the whole framework is derived from the data likelihood. We show that by using the natural correspondence between the Bregman divergences and the exponential families, we can reconstruct the same probabilistic interpretation for the BVDT framework which induces a more general modeling perspective for the problems we consider. The benefit of such probabilistic view is not restricted only to the theoretical aspects; as we show by a walk-through example, one can utilize the probabilistic view of our model to derive appropriate Bregman divergences for those domains where the choice of a good Bregman divergence is not apparent. In particular, in this paper, we use this construction procedure to derive a proper Bregman divergence for frequency data (specifically text data in our experiments). By applying the BVDT framework equipped with the derived divergence on various text datasets, we show the clear advantage of our framework over the original VDT framework in terms of the approximation accuracy, while preserving the same computational complexity. Finally, we show that the original VDT framework is in fact a special case of the BVDT framework.

2 Euclidean Variational Dual-Trees

To prepare the reader for the proposed Bregman divergence extension to the VDT framework, we will in this section briefly review the basic elements of VDT. Interested readers may refer to (Amizadeh et al., 2012a) for further details.

2.1 Motivation

Let $\mathcal{D} = \{x_1, x_2, \ldots, x_N\}$ be a set of i.i.d. datapoints in $\mathcal{X} \subseteq \mathbb{R}^d$. The similarity graph $\mathcal{G} = \langle \mathcal{D}, \mathcal{D} \times \mathcal{D}, \mathbb{W} \rangle$ is defined as a *complete* graph whose nodes are the elements of $\mathcal{D}$ and the matrix $\mathbb{W} = [w_{ij}]_{N \times N}$ represents the edge weights of the graph. A weight $w_{ij} = k(x_i, x_j; \sigma) = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$ is defined by the similarity between nodes x_i and x_j. The closer x_i and x_j are in terms of the Euclidean distance $\|x_i - x_j\|$, the higher the similarity weight w_{ij} will be. The bandwidth parameter σ is a design parameter that scales the similarities in the graph. By abstracting the input space into a graph, the similarity graph essentially captures the geometry of the data.

Once the similarity graph is constructed, one can define a random walk on it. The transition probability distributions are in this case given by the transition matrix $\mathbb{P} = [p_{ij}]_{N \times N}$, where $p_{ij} = w_{ij} / \sum_{k \neq i} w_{ik}$ is the probability of jumping from node x_i to node x_j. Note that the elements in each row of $\mathbb{P}$ sum up to 1 so that each row is a proper probability distribution. The transition matrix is the fundamental quantity used by many graph-based Machine Learning frameworks, as mentioned already in Section 1.

In terms of computational resources, it takes $O(N^2)$ CPU and memory to construct and maintain the transition matrix, which can be problematic when the number of datapoints, N, becomes large. This is, in fact, quite typical in many real-world datasets and therefore leaves us with a serious computational challenge in using many graph-based frameworks. To overcome this challenge, the key idea is to somehow *reduce* the representation of $\mathbb{P}$. The Variational Dual-Tree (VDT) framework provides a non-parametric methodology to approximate and represent

$\mathbb{P}$ in $O(N^{1.5} \log N)$. One big advantage of this framework to its counterparts is that it directly computes the transition matrix without computing the intermediate similarity matrix $\mathbb{W}$. Another advantage of VDT is that given a distance computation of $O(1)$ between any two datapoints, the overall computational complexity of VDT does not depend on the dimensionality d of the input space.

2.2 Variational Dual-Tree Partitioning

The main idea behind the computational reduction by the VDT framework is to *partition* the transition matrix $\mathbb{P}$ into *blocks*, where each block ties the individual transition probabilities in that block into a single number (or parameter). The number of different elements in $\mathbb{P}$ is in this way reduced from N^2 to $|\mathcal{B}|$, the number of blocks. In the VDT framework, a cluster tree hierarchy of the data lets us define blocks of different sizes according to nodes at different granularity levels in the tree. In this way, the number of blocks $|\mathcal{B}|$ can be as small as $O(N)$ (Amizadeh et al., 2012a).

More specifically, let $\mathcal{T}$ be a binary tree that represents a hierarchical clustering of data. Given $\mathcal{T}$, a *valid* block partition $\mathcal{B}$ defines a mutually exclusive and exhaustive partition of $\mathbb{P}$ into blocks (or submatrices) $(A, B) \in \mathcal{B}$, where A and B are two *non-overlapping* subtrees in $\mathcal{T}$. That is, A cannot be a subtree of B or vice versa. A valid block partition $\mathcal{B}$ relates to a similarity graph as follows. If A and B are two non-overlapping datapoint clusters, then the block $(A, B) \in \mathcal{B}$ represents the transition probabilities from datapoints in node A to datapoints in node B with only one parameter, which is denoted by q_{AB}. That is, $\forall x_i \in A, x_j \in B, p_{ij} = q_{AB}$; we call this a *block constraint*. Valid block partitions are not unique; in fact, any further *refinement* of a valid partition results in a new valid partition with increased number of blocks. In the *coarsest* partition, each subtree in $\mathcal{T}$ is blocked with its sibling, resulting in the minimum number of blocks $|\mathcal{B}| = 2(N-1)$. The coarsest partition embodies the approximation of $\mathbb{P}$ at the coarsest level. On the other hand, in the *finest* partition, each leaf in $\mathcal{T}$ is blocked with all the other leaves resulting in the maximum number of blocks $|\mathcal{B}| = N(N-1)$; this partition exactly represents $\mathbb{P}$ with no approximation. The other valid partitions vary between these two extremes. In this setup, a finer partition has better approximation accuracy at the cost of increased computational complexity and vice versa. Figure 1 borrowed from (Amizadeh et al., 2012a) shows an example of a block partitioning with its corresponding cluster tree.

Given a block partitioning $\mathcal{B}$ of $\mathbb{P}$, one needs to compute the parameter set $\mathcal{Q} = \{q_{AB} \mid (A, B) \in \mathcal{B}\}$ as the final step to complete the approximation of $\mathbb{P}$. For this purpose, the VDT framework maximizes the variational lower bound on the log-likelihood of data with the set $\mathcal{Q}$ as the variational parameters. In this framework, the likelihood of data is modeled by the non-parametric kernel density estimate. In particular, a Gaussian kernel is placed on each datapoint in the input space such that each datapoint x_i plays two roles: (I) a datapoint where we want to compute the likelihood at, and (II) the center of a Gaussian kernel. We denote datapoint x_i as m_i when it is regarded as a kernel center. Using this notation, the likelihood of dataset $\mathcal{D}$ is computed as:

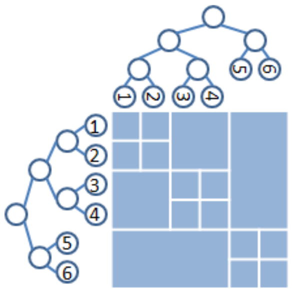


Figure 1: A block partition that, for example, enforces the block constraint $p_{13} = p_{14} = p_{23} = p_{24}$ for the block $(A, B) = (1-2, 3-4)$ (where $a-b$ denotes a through b).

$$p(\mathcal{D}) = \prod_i p(x_i) = \prod_i \sum_{j \neq i} p(m_j) p(x_i \mid m_j), \qquad (1)$$

where $p(x_i|m_j)$ is the Gaussian density at x_i centered at m_j; that is, $p(x_i|m_j) = \exp(-\|x_i - m_j\|^2/2\sigma^2)(2\pi\sigma^2)^{-d/2}$, and $p(m_j) = 1/(N-1)$ is the uniform mixture weight. Using Bayes rule, we observe that the posterior $p(m_j \mid x_i)$ is equal to the transition probability p_{ij}. The lower-bound on the log-likelihood is then computed as:

$$\begin{aligned}\log p(\mathcal{D}) &= \sum_i \log \sum_{j \neq i} \frac{q_{ij}}{q_{ij}} p(m_j) p(x_i \mid m_j) \\ &\geq \sum_i \sum_{j \neq i} q_{ij} \log \frac{p(m_j) p(x_i \mid m_j)}{q_{ij}} \\ &= \log p(\mathcal{D}) - \sum_i D_{KL}(q_{i\cdot} \| p_{i\cdot}) \triangleq \ell(\mathcal{D}), \qquad (2)\end{aligned}$$

where q_{ij}'s are the variational parameters approximating p_{ij}'s and $D_{KL}(q_{i\cdot} \| p_{i\cdot})$ is the KL-divergence between two distributions $q_{i\cdot}$ and $p_{i\cdot}$. With only the sum-to-one constraints $\forall i : \sum_{j \neq i} q_{ij} = 1$, the optimization in Eq. (2) returns $q_{ij} = p_{ij}$; that is, there is no approximation! However, by adding the block constraints from the block partition $\mathcal{B}$, one can rewrite Eq. (2) in terms of the block parameters in $\mathcal{Q}$. Let us first reformulate the sum-to-one constraints in accordance with the block partition as follows:

$$\textstyle\sum_{(A,B)\in\mathcal{B}(x_i)} |B|\cdot q_{AB} = 1 \text{ for all } x_i \in \mathcal{D}. \qquad (3)$$

where, $\mathcal{B}(x_i) \triangleq \{(A,B)\in\mathcal{B} \mid x_i \in A\}$. In this case,

$$\begin{aligned}\ell(\mathcal{D}) &= c - \frac{1}{2\sigma^2}\sum_{(A,B)\in\mathcal{B}} q_{AB}\cdot D_{AB} \\ &\quad - \sum_{(A,B)\in\mathcal{B}} |A||B|\cdot q_{AB}\log q_{AB}, \qquad (4)\end{aligned}$$

where

$$\begin{aligned}c &= -N\log\left((2\pi)^{d/2}\sigma^d(N-1)\right)\\ D_{AB} &= \sum_{x_i\in A}\sum_{m_j\in B}\|x_i - m_j\|^2. \qquad (5)\end{aligned}$$

(Thiesson and Kim, 2012) has proposed an $O(|\mathcal{B}|)$-time algorithm to maximize Eq. (4) under the constraints in Eq. (3).

A very crucial element of the VDT framework is the way that Eq. (5) is computed; the direct computation of the double-sum for D_{AB} of all blocks would send us back to an $O(N^2)$-time algorithm! Fortunately, this can be avoided thanks to the Euclidean distance, where D_{AB} can be written as:

$$D_{AB} = |A|S_2(B) + |B|S_2(A) - 2S_1(A)^TS_1(B), \qquad (6)$$

where, $S_1(A) = \sum_{x\in A} x$ and $S_2(A) = \sum_{x\in A} x^Tx$ are the statistics of subtree A. These statistics can be incrementally computed and stored while the cluster tree is being built; an $O(N)$ computation. Using these statistics, D_{AB} is computed in $O(1)$. The crux of the reformulation in Eq. (6) is the de-coupling of the *mutual interactions* between two clusters A and B so that the sum of mutual interactions can be computed using the sufficient statistics pre-calculated independently for each cluster.

Once the parameters $\mathcal{Q}$ are computed, we have the block approximation of $\mathbb{P}$ which we denote by $\mathbb{Q}$. (Amizadeh et al., 2012a) has proposed an $O(|\mathcal{B}|)$-time algorithm to compute the matrix-vector multiplication $\mathbb{Q}v$ for an arbitrary vector v. We will use this algorithm for label propagation in Eq. (29) in Section 4.2.

2.3 Anchor Tree Construction

So far, we assumed the cluster hierarchy tree $\mathcal{T}$ is given. In reality, however, one needs to efficiently build such a hierarchy from data as the first step of the VDT framework. (Amizadeh et al., 2012a) has used *the anchor construction* method (Moore, 2000) for this purpose. Compared to the classical $O(N^3)$-time agglomerative clustering algorithm, the construction time for this tree is $O(N^{1.5}\log N)$ for a relatively balanced data set (see Amizadeh et al. (2012a), Appendix). The construction starts with an *anchor growing phase* that for the N data points gradually grows a set of $\sqrt{N}$ *anchors* $\mathcal{A}$. Each anchor $A\in\mathcal{A}$ has a *pivot* datapoint A_p and maintains a list of individual *member* datapoints A_M sorted by decreasing order of distance to the pivot $\|x_i - A_p\|$, $x_i \in A_M$. The distance to the first datapoint in the list therefore defines a *covering radius* A_r for the anchor, where $\|x_i - A_p\| \le A_r$ for all $x_i \in A_M$.

The anchor growing phase constructs a first anchor by choosing a random datapoint as pivot and assigning all datapoints as members of that anchor. A new anchor A^{new} is now added to a current set of anchors in three steps until $\sqrt{N}$ anchors are found: first, its pivot element is chosen as the datapoint with largest distance to the pivot(s) of the current anchor(s):

$$A_p^{new} = \arg\max_{x_i\in A_M, A\in\mathcal{A}} \|x_i - A_p\|. \qquad (7)$$

Second, the new anchor now iterates through the member elements of current anchors and *"steals"* the datapoints with $\|x_i - A_p^{new}\| < \|x_i - A_p\|$. Because the list of elements in an anchor is sorted with respect to $\|x_i - A_p\|$, a significant computational gain is achieved by not evaluating the remaining datapoints in the list once we discover that for the i-th datapoint in the list $\|x_i - A_p\| \le d_{thr}$, where

$$d_{thr} = \|A_p^{new} - A_p\|/2, \qquad (8)$$

This guarantees that the elements with index $j \ge i$ in the list are closer to their original anchor's pivot and cannot be stolen by the new anchor. When a new anchor is done stealing datapoints from older anchors, its list of elements is finally sorted.

Once the $\sqrt{N}$ anchors are created, the anchor tree construction now proceeds to *anchor agglomeration phase* that assigns anchors as leaf nodes and then iteratively merges two nodes that create a parent node with the *smallest* covering radius. This agglomerative bottom-up process continues until a hierarchical binary tree is constructed over the $\sqrt{N}$ initial anchors. With an Euclidean distance metric, the covering parent for two nodes A and B can be readily computed as the node C with pivot and radius

$$\begin{aligned}C_p &= (|A|\cdot A_p + |B|\cdot B_p)/(|A|+|B|) \qquad (9)\\ C_r &= (A_r + B_r + \|B_p - A_p\|)/2 \qquad (10)\end{aligned}$$

Finally, recall that the leaves (i.e. the initial anchors) in this newly constructed tree contain $\sqrt{N}$ datapoints on average each. The whole construction algorithm is now recursively called for each anchor leaf to grow it into a subtree. The recursion ends when the leaves of the tree contain only one datapoint each.

3 Bregman Variational Dual-Trees

The Euclidean VDT framework has been shown to be a practical choice for large-scale applications. However, its inherent assumption that the underlying distance metric in the input space should be the Euclidean distance is somewhat restrictive. In many real-world problems, the Euclidean distance is simply not the best way to quantify the similarity between datapoints.

On the other hand, the VDT framework does not seem to depend on the choice of distance metric, which makes it very tempting to replace the Euclidean distance in the formulation of the VDT framework with a general distance metric. However, there is one problem: the de-coupling in Eq. (6) was achieved only because of the Euclidean distance special form which is not the case for a general distance metric. Unfortunately, we cannot compromise on this de-coupling simply because, without it, the overall complexity of the framework is back to $O(N^2)$. One solution is to use some approximation technique similar to Fast Gauss Transform techniques (Yang et al., 2005) to approximately de-couple a general distance metric. Although, this may work well for some special cases, in general, the computational burden of such approximation can be prohibitive; besides, we will have a new source of approximation error.

So, if we cannot extend VDT for general distance metric, is there any sub-class of metrics or divergences which we can safely use to extend the VDT framework? The answer is yes, the family of *Bregman divergences* is a qualified candidate. This family contains a diverse set of divergences which also include the Euclidean distance. By definition, the Bregman divergence has a de-coupled form which makes it perfect for our purpose. From the applied side, Bregman divergences cover some very practical divergences and metrics such as Euclidean distance, KL-Divergence and Logistic Loss that are widely used in many engineering and scientific applications. Furthermore, the natural correspondence of Bregman divergences with the exponential families provides a neat probabilistic interpretation for our framework.

3.1 Bregman Divergence and The Exponential Families

Before illustrating the Bregman Variational Dual-Tree (BVDT) framework, we briefly review the Bregman divergence, its important properties and its connection to the exponential families. Interested readers may refer to (Banerjee et al., 2005) for further details.

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be a convex set in $\mathbb{R}^d$, $ri(\mathcal{X})$ denote the relative interior of $\mathcal{X}$, and $\phi : \mathcal{X} \mapsto \mathbb{R}$ be a strictly convex function differentiable on $ri(\mathcal{X})$, then the Bregman divergence $d_\phi : \mathcal{X} \times ri(\mathcal{X}) \mapsto [0, \infty)$ is defined as:

$$d_\phi(x, y) \triangleq \phi(x) - \phi(y) - (x - y)^T \nabla\phi(y) \tag{11}$$

where, $\nabla\phi(y)$ is the gradient of $\phi(\cdot)$ evaluated at y. For different choices of $\phi(\cdot)$, we will get different Bregman divergences. Table 1 lists some famous Bregman divergences along with their corresponding $\phi(\cdot)$ functions. It is important to note that the general Bregman divergence is *not* a distance metric: it is not symmetric, nor does it satisfy the triangular inequality. However, we have $\forall x \in \mathcal{X}, y \in ri(\mathcal{X}) : d_\phi(x, y) \geq 0, d_\phi(y, y) = 0$.

Let $\mathcal{S} = \{x_1, \dots, x_n\} \subset \mathcal{X}$ and X be a random variable that takes values from $\mathcal{S}$ with uniform distribution[1]; the *Bregman information* of the random variable X for the Bregman divergence $d_\phi(\cdot, \cdot)$ is defined as:

$$I_\phi(X) \triangleq \min_{s \in ri(\mathcal{X})} \mathbb{E}\big[d_\phi(X, s)\big] = \min_{s \in ri(\mathcal{X})} \frac{1}{n} \sum_{i=1}^{n} d_\phi(x_i, s) \tag{12}$$

The optimal s that minimizes Eq. (12) is called the *Bregman representative* of X and is equal to:

$$\arg\min_{s \in ri(\mathcal{X})} \mathbb{E}\big[d_\phi(X, s)\big] = E[X] = \frac{1}{n} \sum_{i=1}^{n} x_i \triangleq \mu \tag{13}$$

That is, the Bregman representative of X is always equal to the sample mean of $\mathcal{S}$ *independent of the Bregman divergence* $d_\phi(\cdot, \cdot)$.

The probability density function $p(z)$, defined on set $\mathcal{Z}$, belongs to an *exponential family* if there exists a mapping $g : \mathcal{Z} \mapsto \mathcal{X} \subseteq \mathbb{R}^d$ that can be used to reparameterize $p(z)$ as:

$$p(z) = p(x; \theta) = \exp(\theta^T x - \psi(\theta)) p_0(x) \tag{14}$$

where $x = g(z)$ is the *natural statistics* vector, θ is the *natural parameter* vector and $\psi(\theta)$ is the *log-partition function.* Eq. (14) is called the *canonical form* of p. If θ takes values from parameter space Θ, Eq. (14) defines the family $\mathcal{F}_\psi = \{p(x; \theta) \mid \theta \in \Theta\}$ parameterized by θ. If Θ is an open set and we have that $\nexists c \in \mathbb{R}^d$ s.t. $c^T g(z) = 0, \forall z \in \mathcal{Z}$, then family $\mathcal{F}_\psi$ is called a *regular exponential family.* (Banerjee et al., 2005) has shown that any probability density function $p(x; \theta)$ of a regular exponential family with the canonical form of Eq. (14) can be uniquely expressed as:

$$p(x; \theta) = \exp(-d_\phi(x, \mu)) \exp(\phi(x)) p_0(x) \tag{15}$$

where $\phi(\cdot)$ is the *conjugate function* of the log-partition function $\psi(\cdot)$, $d_\phi(\cdot, \cdot)$ is the Bregman divergence defined w.r.t. function $\phi(\cdot)$, and μ is the *mean parameter.* The mean parameter vector μ and the natural

[1]The results hold for any distribution on $\mathcal{S}$.

parameter vector θ are connected through:

$$\mu = \nabla\psi(\theta), \ \theta = \nabla\phi(\mu) \tag{16}$$

Moreover, (Banerjee et al., 2005) (Theorem 6) has shown there is a bijection between the regular exponential families and the regular Bregman divergences. The last column in Table 1 shows the corresponding exponential family of each Bregman divergence. Using Eq. (13), one can also show that, given the finite sample $\mathcal{S} = \{x_1, \ldots, x_n\}$, the maximum-likelihood estimate of the mean parameter $\hat{\mu}$ for any regular exponential family $\mathcal{F}_\psi$ is always equal to the sample mean of $\mathcal{S}$ regardless of $\mathcal{F}_\psi$.

3.2 Bregman Variational Approximation

Having described the basic concepts of Bregman divergences, we are now ready to nail down the BVDT framework. Let $\mathcal{S} = \{z_1, z_2, \ldots, z_N\} \subset \mathcal{Z}$ be a finite sample from the convex set $\mathcal{Z}$ which is not necessarily an Euclidean space. We are interested to approximate the transition matrix $\mathbb{P}$ on the similarity graph of $\mathcal{S}$ where we know the Euclidean distance is not necessarily the best way to encode similarity. To do so, we assume $\mathcal{S}$ is sampled according to an unknown mixture density model $p^*(z)$ with K components from the regular exponential family $\mathcal{F}_\psi$. That is, there exists the mapping $g : \mathcal{Z} \mapsto \mathcal{X} \subseteq \mathbb{R}^d$ such that $p^*(z)$ can be re-parametrized in the canonical form as $p^*(x) = \sum_{i=1}^K p(\theta_i)\exp(\theta_i^T x - \psi(\theta_i))p_0(x)$.[2]

Furthermore, let $\mathcal{D} = g(\mathcal{S}) = \{x_1, x_2, \ldots, x_N\} \subset \mathcal{X}$ be the natural statistics of sample $\mathcal{S}$ s.t. $x_i = g(z_i)$. Then we model the likelihood of $\mathcal{D}$ using the kernel density estimation:

$$p(\mathcal{D}) = \prod_{i=1}^N \sum_{j\neq i} p(m_j)p(x_i \mid m_j) \tag{17}$$

$$= \prod_{i=1}^N \sum_{j\neq i} p(m_j)\exp(-d_\phi(x_i, m_j))\exp(\phi(x_i))p_0(x_i)$$

Where, we assume the kernel component $p(x_i \mid m_j)$ belongs to the regular exponential family $\mathcal{F}_\psi$ such that it can be uniquely re-parameterized using Eq. (15). Given a block partitioning $\mathcal{B}$ on $\mathbb{P}$, we follow the similar steps in Eq. (2)-(5) to derive the block-partitioned variational lower-bound on $p(\mathcal{D})$:

$$\ell(\mathcal{D}) = c - \sum_{(A,B)\in\mathcal{B}} q_{AB}\cdot D_{AB} - \sum_{(A,B)\in\mathcal{B}} |A||B|\cdot q_{AB}\log q_{AB}, \tag{18}$$

[2]For some exponential families such as Gaussian and Multinomial, the mapping $g(\cdot)$ is identity.

where

$$c = -N\log(N-1) + \sum_{i=1}^N \big(\phi(x_i) + \log p_0(x_i)\big)$$

$$D_{AB} = \sum_{x_i\in A}\sum_{m_j\in B} d_\phi(x_i, m_j). \tag{19}$$

Now we can maximize $\ell(\mathcal{D})$ subject to the constraints in Eq. (3) to find the approximation $\mathbb{Q}$ of $\mathbb{P}$ using the same $O(|\mathcal{B}|)$-time algorithm in the VDT framework.

The crucial aspect of the BVDT framework is D_{AB} in Eq. (19) is de-coupled into statistics of the subtrees A and B using the definition of the Bregman divergence:

$$D_{AB} = |B|S_1(A) + |A|\big(S_2(B) - S_1(B)\big) - S_3(A)^T S_4(B) \tag{20}$$

where,

$$S_1(A) = \sum_{x\in A}\phi(x), \qquad S_2(A) = \sum_{x\in A} x^T\nabla\phi(x)$$

$$S_3(A) = \sum_{x\in A} x, \qquad S_4(A) = \sum_{x\in A}\nabla\phi(x) \tag{21}$$

are the statistics of subtree A. These statistics can be incrementally computed and stored while the cluster tree is being built (in overall $O(N)$ time) such that at the optimization time, D_{AB} is computed in $O(1)$.

Finally, by setting $\phi(x) = \|x\|^2/2\sigma^2$ and doing the algebra, the BVDT framework reduces to the Euclidean VDT framework; that is, the Euclidean VDT framework is a special case of the BVDT framework.

3.3 Bregman Anchor Trees

Recall that the approximation in the Euclidean VDT framework is based on the cluster hierarchy $\mathcal{T}$ of the data which is built using the anchor tree method with the Euclidean distance. For the BVDT framework, we can no longer use this algorithm because the Euclidean distance no longer reflects the similarity in the input space. For this reason, one needs to develop an anchor tree construction algorithm for general Bregman divergences. This generalization is not straightforward as a general Bregman divergence is neither symmetric nor does it hold the triangle inequality. In particular, we need to address two major challenges.

First, due to the asymmetry of a general Bregman divergence, the merging criterion in the anchor agglomeration phase is no longer meaningful. For this purpose, we have used the criterion suggested in the recent work by (Telgarsky and Dasgupta, 2012). In particular, at each agglomeration step, anchors A and B with the minimum *merging cost* are picked to merge into

Name	$\mathcal{X}$	$\boldsymbol{\phi}(\mathbf{x})$	$\mathbf{d}_{\boldsymbol{\phi}}(\mathbf{x}, \mathbf{y})$	**Exponential Family**
Logistic Loss	$(0, 1)$	$x \log x$	$x \log\left(\frac{x}{y}\right) + (1-x)\log\left(\frac{1-x}{1-y}\right)$	1D Bernoulli
Itakura-Saito Dist.	$\mathbb{R}_{++}$	$-\log x - 1$	$\frac{x}{y} - \log\left(\frac{x}{y}\right) - 1$	1D Exponential
Relative Entropy	$\mathbb{Z}_{+}$	$x \log x - x$	$x \log\left(\frac{x}{y}\right) - x + y$	1D Poisson
Euclidean Dist.	$\mathbb{R}^d$	$\|x\|^2/2\sigma^2$	$\|x-y\|^2/2\sigma^2$	Spherical Gaussian
Mahalonobis Dist.	$\mathbb{R}^d$	$x^T\Sigma^{-1}x$	$(x-y)^T\Sigma^{-1}(x-y)$	Multivariate Gaussian
KL-Divergence	d-simplex	$\sum_{j=1}^{d} x(j)\log x(j)$	$\sum_{j=1}^{d} x(j)\log\left(\frac{x(j)}{y(j)}\right)$	-
-	int. d-simplex	$\sum_{j=1}^{d} x(j)\log\left(\frac{x(j)}{L}\right)$	$\sum_{j=1}^{d} x(j)\log\left(\frac{x(j)}{y(j)}\right)$	Multinomial

Table 1: Famous Bregman divergences along with their corresponding $\phi(\cdot)$ function, its domain and the corresponding exponential family distribution

the parent anchor C. The cost for merging a pair of anchors A and B is defined as:

$$\Delta(A, B) = |A| \cdot d_\phi(A_p, C_p) + |B| \cdot d_\phi(B_p, C_p) \quad (22)$$

where $|A|$ is the number of elements in the anchor A, and C_p is the parent anchor's pivot which is given by Eq. (9). (Telgarsky and Dasgupta, 2012) has shown that the merging cost in Eq. (22) can be interpreted as the difference of cluster unnormalized Bregman informations before and after merging.

Second, as shown before, using the halfway Euclidean distance as the stealing threshold (Eq. (8)) in the anchor construction phase significantly cuts the unnecessary computations. This threshold, however, is meaningless for a general Bregman divergence simply because a general Bregman divergence is not a metric. Therefore, we need to develop an equivalent threshold for Bregman divergences to achieve a similar computational gain in constructing Bregman anchor trees. The following proposition addresses this problem:

Preposition 1. *Let A^{curr} and A^{new} denote the current and the newly created anchors, respectively, where A^{new} is stealing datapoints from A^{curr}. Define*

$$d_{thr} = \frac{1}{2}\min_{y \in \mathcal{X}}\left[d_\phi(y, A_p^{curr}) + d_\phi(y, A_p^{new})\right] \quad (23)$$

Then for all $x \in A^{curr}$ such that $d_\phi(x, A_p^{curr}) \le d_{thr}$, we will have $d_\phi(x, A_p^{curr}) \le d_\phi(x, A_p^{new})$; that is, x cannot be stolen from A_p^{curr} by A_p^{new}. Furthermore, the minimizer of Eq. (23) is equal to:

$$y^* = \nabla\phi^{-1}\left[\frac{1}{2}\Big(\nabla\phi(A_p^{curr}) + \nabla\phi(A_p^{new})\Big)\right] \quad (24)$$

The proof is easily derived by contradiction. Note that for the special case of Euclidean distance where $\phi(x) = \|x\|^2/2\sigma^2$, Eq. (23) reduces to Eq. (8).

4 Experiments

In order to apply the BVDT framework to a real problem, all one needs to know are the appropriate Bregman divergence and its corresponding $\phi(\cdot)$ function in the input space; knowing the corresponding exponential family that has generated the data is *not* necessary. However, in many problems, the choice of the appropriate Bregman divergence is not clear; instead, we know that the underlying data generation process belongs to *or* can be accurately modeled with exponential families. In such cases, one can systematically derive the appropriate Bregman divergence from the data distribution. In this section, we use this procedure to derive an appropriate Bregman divergence for the frequency data.

4.1 Modeling Frequency Data

Given a set of d events $\{e_j\}_{j=1}^d$, the frequency dataset $\mathcal{D}$ consists of N feature vectors where the j-th element of the i-th vector, $x_i(j)$, represents the number of times that event e_j happens in the i-th case for all $j \in \{1, \dots, d\}$. The length of each vector is defined as the total number of events happened in that case, i.e. $L_i = \sum_{j=1}^d x_i(j)$. For example, in text analysis, events can represent all the terms that appeared in a text corpus, while the data cases represent the documents. The frequency dataset in this case is the famous bag-of-words. We adopt the term-document analogy for the rest of this section.

If the lengths of all documents were equal, one could model the document generation process with a mixture of multinomials, where each mixture component had different term generation probabilities while sharing the same length parameter (Banerjee et al., 2005). However, having the same length is not the case for many real datasets. To address this issue, we propose a mixture model for data generation whose k-th component ($k \in [1..K]$) is modeled by the following generative model:

$$p_k(x; \alpha_k, \lambda_k) = p(x \mid L; \boldsymbol{\alpha}_k)p(L; \lambda_k) \quad (25)$$

where, the length of a document, L, has a poisson distribution $p(L; \lambda_k)$ with mean length λ_k and given L, document x has a multinomial distribution $p(x \mid$

$L; \boldsymbol{\alpha}_k)$ with the term probabilities $\boldsymbol{\alpha}_k = [\alpha_k(j)]_{j=1}^d$. By doing the algebra, $p_k(x; \alpha_k, \lambda_k)$ can be written in the form of Eq. (14), where we have:

$$\theta_k = [\theta_k(j)]_{j=1}^d = \left[\log(\lambda_k \alpha_k(j))\right]_{j=1}^d,$$
$$\psi(\theta_k) = \sum_{j=1}^d \exp(\theta_k(j)), p_0(x) = \left(\prod_{j=1}^d x(j)!\right)^{-1} \quad (26)$$

That is, our generative model also belongs to an exponential family. By deriving the conjugate function of $\psi(\cdot)$, we get the $\phi(\cdot)$ function and its corresponding Bregman divergence as:

$$\phi(x) = \sum_{j=1}^d x(j)\log x(j) - \sum_{j=1}^d x(j) \quad (27)$$
$$d_\phi(x, y) = \sum_{j=1}^d \left[x(j)\log\left(\frac{x(j)}{y(j)}\right) - x(j) + y(j)\right] \quad (28)$$

The divergence in Eq. (28) is called the *Generalized I-Divergence* (GID) which is a generalization of KL-Divergence (Dhillon and Sra, 2005). As a result, to work with frequency data (in particular text data in our experiments), we customize the BVDT framework to use GID as its Bregman divergence. It should be noted that GID is not the only non-Euclidean similarity measure between documents, other techniques such as co-citation (Šingliar and Hauskrecht, 2006) has been used before.

4.2 Experimental Setup

To evaluate the BVDT framework, we use it for a semi-supervised learning (SSL) task on various text datasets. (Amizadeh et al., 2012a) has experimentally shown the VDT framework can be well scaled to large-scale problems. Our framework has exactly the same order of complexity as VDT. Therefore, due to the space limit, we focus our evaluation only on the *accuracy* of SSL for text data. In particular, we show that while enjoying the same computational speed-up as the Euclidean VDT framework, the BVDT framework equipped with GID significantly improves the quality of learning over the VDT framework for text data.

For the SSL task, we want to *propagate* the labels from a small set of labeled examples to the rest of unlabeled examples over the similarity graph built over the examples. To do so, we use the following iterative propagation scheme (Zhou et al., 2003):

$$y^{(t+1)} \leftarrow \alpha \mathbb{M} y^{(t)} + (1-\alpha)y^0 \quad (29)$$

where $y^{(t)} \in \mathbb{R}^{N\times 1}$ is the vector of labels at time t, $\mathbb{M} \in \mathbb{R}^{N\times N}$ is the transition matrix $\mathbb{P}$ (or its approximation $\mathbb{Q}$), y^0 is the vector of initial partial labeling and $\alpha \in [0, 1]$ is the mixing coefficient. For all of our experiments, we iterate through this process 300 times with $\alpha = 0.01$. Upon completion of propagation, we compute the classification accuracy over the unlabeled data w.r.t. the held-out true labels. Note that this process is for binary classification problems; for problems with multiple classes, we perform one-vs-all scheme for each class and take the maximum.

We have compared four methods for computing matrix $\mathbb{M}$: (a) Euclidean VDT, (b) BVDT equipped with GID, (c) Exact method with Euclidean distance and (d) Exact method with GID. For Exact methods, the transition matrix $\mathbb{P}$ is exactly computed using the direct method. Due to memory limitations on our single machine, we could apply Exact methods only for the smaller datasets. For variational methods, we have used the coarsest level of approximation for the transition matrix; that is, $\mathbb{P}$ is approximated with only $2(N-1)$ number of parameters (or blocks).

We have applied these methods on five text datasets represented as bag-of-words used before in (Greene and Cunningham, 2006; Deng et al., 2011; Maas et al., 2011; Lang, 1995). Table 2 illustrates the details.

Dataset	N	d	C
BBC-Sport News Articles	737	4613	5
BBC News Articles	2225	9636	5
20 Newsgroup	11269	61188	20
NSF Research Abstracts	16405	18674	10
Large Movie Reviews	50000	89527	2

Table 2: Datasets used: N = number of documents, d = number of terms, C = number of classes

4.3 Results

Figures 2(A-E) show the classification accuracy vs. the percentage of labeled data for the datasets. The plots show the average of 5 trials with 95% confidence intervals. Although, none of the actual datasets is generated by the generative model proposed in Eq. (25), the BVDT with GID method consistently and significantly outperforms the Euclidean VDT. In particular, these results show that (a) the Generalized I-Divergence derived from the proposed generative model for text data captures the document similarity much better than the Euclidean distance does, and (b) the BVDT framework provides a straightforward mechanism to extend the variational dual-tree method beyond the Euclidean distance to use Bregman divergences such as GID.

We have also applied the Exact methods to the two smallest datasets. Not surprisingly, the Exact method with GID has the best performance compared to other methods. However, the Exact method with a wrong distance metric (the Euclidean distance in this case) can do even worse than the VDT method with Euclidean distance, as observed for the second BBC

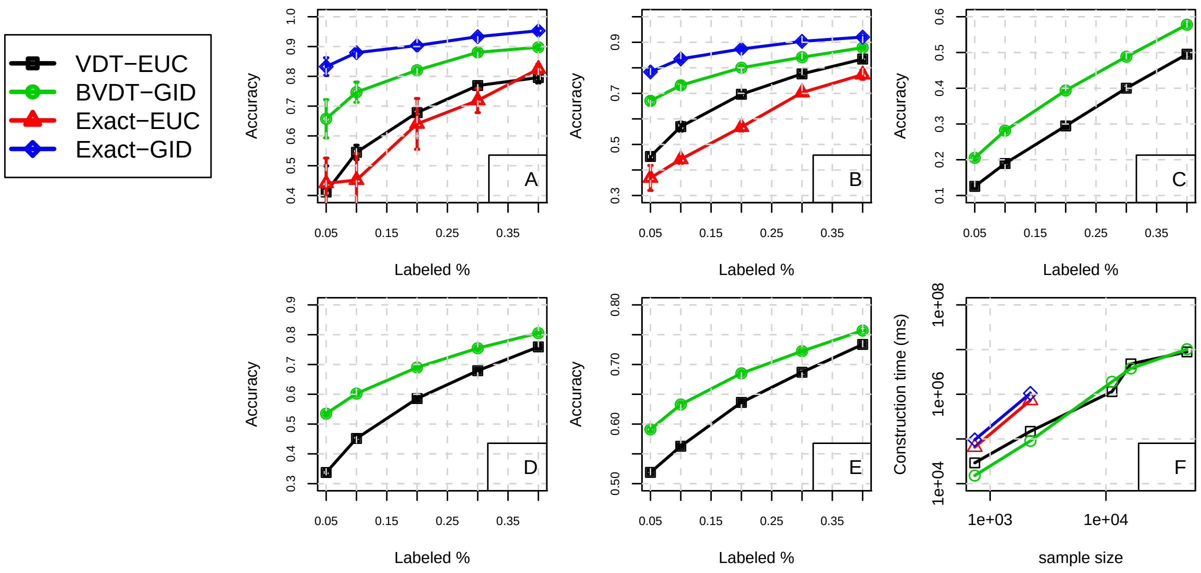


Figure 2: The accuracy curves vs. labeled data % for (A) BBC Sport News (B) BBC News (C) 20 Newsgroup (D) NSF Research Abstracts (E) Large Movie Reviews. (F) The computational complexity of four methods vs. the dataset size

dataset. We conjecture the existence of block regularization in the VDT framework compensates for the improper distance to some degree in this case.

Finally, we note the computational complexity of the aforementioned methods vs. the dataset size (i.e. the number of documents) shown in Figure 2(F). Both X an Y axes in this plot are in log-scale. As the plot shows while Euclidean VDT and BVDT have the same order complexity, they both are orders of magnitude faster than the Exact methods. In other words, while significantly improving on learning accuracy, BVDT still enjoys the same computational benefits as VDT.

5 Conclusions

In this paper, we proposed the Bregman Variational Dual-Tree framework which is the generalization of the recently developed Euclidean Variational Dual-Tree method to Bregman divergences. The key advantage of the BVDT framework is it covers a large class of distances and divergences and therefore makes the variational dual-trees accessible to many non-Euclidean large-scale datasets. The crucial aspect of our generalization to Bregman divergences is, unlike generalizing VDT to a general distance metric, it comes with no extra computational cost; that is, its computational order can still be kept the same as that of the VDT framework. This is very important to the development of whole framework since the variational dual-trees were originally developed to tackle large-scale problems. To achieve this, we utilized the functional form of the general Bregman divergence to design a bottom-up mechanism to cut unnecessary distance computations similar to that of the Euclidean VDT framework.

Furthermore, by exploiting the connection between the Bregman divergences and the exponential families, we provided a probabilistic view of our model. By a walk-through example, we showed that this probabilistic view can be used to derive the appropriate Bregman divergence for those domains where the best choice of distance in not apparent at the first glance. This example provides us with a powerful construction procedure to develop the appropriate Bregman divergence for a given problem. Specifically, we used this procedure to derive the Generalized I-Divergence for the frequency data. We showed that by incorporating GID in the BVDT framework, our model significantly improved the accuracy of learning for semi-supervised learning on various text datasets, while maintaining the same order of complexity as the original VDT framework. It should be emphasized that although we used the BVDT framework with one type of Bregman divergence, the proposed model is general and can be customized with any Bregman divergence.

Acknowledgements

This research work was supported by grants 1R01LM010019-01A1 and 1R01GM088224-01 from the NIH. The content of this paper is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

S. Amizadeh, S. Wang, and M. Hauskrecht. An efficient framework for constructing generalized locally-induced text metrics. In *IJCAI-11*, pages 1159–1164, 2011.

S. Amizadeh, B. Thiesson, and M. Hauskrecht. Variational dual-tree framework for large-scale transition matrix approximation. In *the 28th Conference on Uncertainty in Artificial Intelligence (UAI-12)*, pages 64–73, 2012a.

S. Amizadeh, H. Valizadegan, and M. Hauskrecht. Factorized diffusion map approximation. *Journal of Machine Learning Research-Proceedings Track*, 22: 37–46, 2012b.

A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. In *ACM SIGKDD*, pages 509–514, 2004.

A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with bregman divergences. *JMLR*, 6: 1705–1749, 2005.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, 2002.

L. Cayton. Fast nearest neighbor retrieval for bregman divergences. In *ICML*, pages 112–119, 2008.

H. Deng, J. Han, B. Zhao, Y. Yu, and C. X. Lin. Probabilistic topic models with biased propagation on heterogeneous information networks. In *KDD*, pages 1271–1279, 2011.

I. Dhillon and S. Sra. *Generalized nonnegative matrix approximations with Bregman divergences*. 2005.

D. Greene and P. Cunningham. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *ICML*, pages 377–384, 2006.

T. Jebara, J. Wang, and S. Chang. Graph construction and b-matching for semi-supervised learning. In *ICML*, volume 382, page 56. ACM, 2009.

S. Kumar, M. Mohri, and A. Talwalkar. Sampling techniques for the nystrom method. *Journal of Machine Learning Research - Proceedings Track*, 5:304–311, 2009.

Ken Lang. News weeder: Learning to filter netnews. In *Machine Learning International Workshop*, pages 331–339. Morgan Kufmann Publishers, Inc., 1995.

D. Lee, A.G. Gray, and A.W. Moore. Dual-tree fast gauss transforms. *arXiv preprint arXiv:1102.2878*, 2011.

A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proc. of 49th An. Meeting of the Assoc. for Comp. Ling.*, pages 142–150, June 2011.

A. W. Moore. The anchors hierarchy: Using the triangle inequality to survive high dimensional data. In *UAI*, pages 397–405, 2000.

A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, 2001a.

A. Y. Ng, A. X. Zheng, and M. I. Jordan. Stable algorithms for link analysis. In *SIGIR '01*, pages 258–266, August 2001b.

A. Y. Ng, A. X. Zheng, and M. I. Jordan. Link analysis, eigenvectors and stability. *IJCAI*, pages 903–910, 2001c.

L. Qiao, S. Chen, and X. Tan. Sparsity preserving projections with applications to face recognition. *Pattern Recognition*, 43(1):331–341, 2010.

T Šingliar and M. Hauskrecht. Noisy-or component analysis and its application to link analysis. *The Journal of Machine Learning Research*, 7:2189–2213, 2006.

A. Talwalkar, S. Kumar, and H. A. Rowley. Large-scale manifold learning. In *CVPR*, pages 1–8, 2008.

M. Telgarsky and S. Dasgupta. Agglomerative bregman clustering. *arXiv:1206.6446*, 2012.

B. Thiesson and J. Kim. Fast variational mode-seeking. In *AISTATS 2012, JMLR 22: W&CP 22*. Journal of Machine Learning Research, 2012.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

C. Yang, R. Duraiswami, N.A. Gumerov, and L. Davis. Improved fast gauss transform and efficient kernel density estimation. In *Computer Vision*, pages 664–671, 2003.

C. Yang, R. Duraiswami, L. Davis, et al. Efficient kernel machines using the improved fast gauss transform. *NIPS*, 17:1561–1568, 2005.

L. Zhang, S. Chen, and L. Qiao. Graph optimization for dimensionality reduction with sparsity constraints. *Pattern Recognition*, 45(3):1205–1210, 2012.

Z. Zhang, B. C. Ooi, S. Parthasarathy, and A. KH Tung. Similarity search on bregman divergence: Towards non-metric indexing. *Proc. of the VLDB Endowment*, 2(1):13–24, 2009.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*. MIT Press, 2003.

X. Zhu. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2005.

Hinge-loss Markov Random Fields: Convex Inference for Structured Prediction

Stephen H. Bach **Bert Huang** **Ben London** **Lise Getoor**
Computer Science Dept.
University of Maryland
College Park, MD, 20742, USA

Abstract

Graphical models for structured domains are powerful tools, but the computational complexities of combinatorial prediction spaces can force restrictions on models, or require approximate inference in order to be tractable. Instead of working in a combinatorial space, we use hinge-loss Markov random fields (HL-MRFs), an expressive class of graphical models with log-concave density functions over continuous variables, which can represent confidences in discrete predictions. This paper demonstrates that HL-MRFs are general tools for fast and accurate structured prediction. We introduce the first inference algorithm that is both scalable and applicable to the full class of HL-MRFs, and show how to train HL-MRFs with several learning algorithms. Our experiments show that HL-MRFs match or surpass the predictive performance of state-of-the-art methods, including discrete models, in four application domains.

1 INTRODUCTION

The study of probabilistic modeling in structured and relational domains primarily focuses on predicting discrete variables [12, 19, 24]. However, except for some isolated cases, the combinatorial nature of discrete, structured prediction spaces requires concessions: most notably, for inference algorithms to be tractable, they must be relaxed or approximate (e.g., [19, 22, 25]), or the model's structure must be restricted (e.g., [2, 8]). Broecheler et al. [6] introduced a class of models for continuous variables that has the potential to combine fast and exact inference with the expressivity of discrete graphical models, but this potential has not been well explored. Now called *hinge-loss Markov random fields* (HL-MRFs) [3], these models are analogous to discrete Markov random fields, except that random variables are continuous valued in the unit interval [0,1], and potentials are linear or squared hinge-loss functions.

In this work, we demonstrate that HL-MRFs are powerful tools for structured prediction by producing state-of-the-art performance in a number of domains. We are the first to leverage some of the most powerful features of HL-MRFs, such as squared potentials, which we show produce better results on multiple tasks. We show that HL-MRFs are well-suited to structured prediction for the following reasons. They are expressive, interpretable, and easily defined using the modeling language *probabilistic soft logic* (PSL) [6, 13]. Further, continuous variables are useful both for modeling continuous data as well as for expressing confidences in discrete predictions. Confidences are desirable for the same reason that practitioners often prefer marginal probabilities to the single most probable discrete prediction. Finally, HL-MRFs have log-concave density functions, so finding an exact *most probable explanation* (MPE) for a given input is a convex optimization, and therefore exactly solvable in polynomial time.

Our specific contributions include the following. First, we introduce a new, fast algorithm for MPE inference in HL-MRFs, which is the first to be both scalable and applicable to the full class of HL-MRFs. Second, we show how to train HL-MRFs with several learning algorithms. Third, we show empirically that these advances enable HL-MRFs to tackle a diverse set of relational and structured prediction tasks, providing state-of-the-art performance on collective classification, social-trust prediction, collaborative filtering, and image reconstruction. In particular, we show that HL-MRFs can outperform their discrete counterparts, as well as other leading methods.

1.1 RELATED WORK

Probabilistic soft logic (PSL) [6, 13] is a declarative language for defining templated HL-MRFs. Its development was partially motivated by the need for rich models of continuous similarity values, for use in tasks such as entity resolution, collective classification, and ontology alignment. PSL is in a family of systems for defining templated, relational probabilistic models that includes, for instance, *Markov logic networks* [19], *relational dependency networks* [17], and *relational Markov networks* [23]. In our experiments, we compare against Markov logic networks, which use a first-order syntax similar to PSL's to build discrete probabilistic models.

MPE inference algorithms for HL-MRFs solve a constrained, convex optimization. A standard approach for general, constrained, convex optimization is to use an interior-point method, which Broecheler et al. [6] use. While theoretically efficient, the practical running time of interior-point optimizers quickly becomes cumbersome for large problems. For discrete graphical models, recent advances use *consensus optimization* to obtain fast, approximate MPE inference algorithms [5, 15, 16]. Bach et al. [3] recently developed an analogous algorithm for exact MPE inference in HL-MRFs that produced a significant improvement in running time over interior-point methods, though it was limited to pairwise potentials and constraints, and cost considerably more computation to optimize over squared potentials.

The learning methods we adapt for HL-MRFs are standard approaches for learning parameters of probabilistic models. In particular, our adaptations are analogous to previous learning algorithms for relational and structured models using approximate maximum-likelihood or maximum-pseudolikelihood estimation [6, 14, 17, 19] and large-margin estimation [11, 12, 24].

2 HINGE-LOSS MARKOV RANDOM FIELDS

Hinge-loss Markov random fields (HL-MRFs) are a general class of conditional, continuous probabilistic models [3]. HL-MRFs are log-linear models whose features are hinge-loss functions of the variable states. Through constructions based on *soft logic*, hinge-loss potentials can be used to model generalizations of logical conjunction and implication. HL-MRFs can be defined using the modeling language *probabilistic soft logic* (PSL) [6, 13], making these powerful models interpretable, flexible, and expressive. In this section, we formally present constrained hinge-loss energy functions, HL-MRFs, and briefly review PSL.

Definition 1. *Let $\mathbf{Y} = (Y_1, \ldots, Y_n)$ be a vector of n variables and $\mathbf{X} = (X_1, \ldots, X_{n'})$ a vector of n' variables with joint domain $\mathbf{D} = [0,1]^{n+n'}$. Let $\phi = (\phi_1, \ldots, \phi_m)$ be m continuous potentials of the form*

$$\phi_j(\mathbf{Y}, \mathbf{X}) = [\max\{\ell_j(\mathbf{Y}, \mathbf{X}), 0\}]^{p_j}$$

where ℓ_j is a linear function of $\mathbf{Y}$ and $\mathbf{X}$ and $p_j \in \{1, 2\}$. Let $C = (C_1, \ldots, C_r)$ be linear constraint functions associated with index sets denoting equality constraints $\mathcal{E}$ and inequality constraints $\mathcal{I}$, which define the feasible set

$$\tilde{\mathbf{D}} = \left\{ \mathbf{Y}, \mathbf{X} \in \mathbf{D} \,\middle|\, \begin{array}{l} C_k(\mathbf{Y}, \mathbf{X}) = 0, \forall k \in \mathcal{E} \\ C_k(\mathbf{Y}, \mathbf{X}) \geq 0, \forall k \in \mathcal{I} \end{array} \right\}.$$

For $\mathbf{Y}, \mathbf{X} \in \tilde{\mathbf{D}}$, given a vector of nonnegative free parameters, i.e., weights, $\lambda = (\lambda_1, \ldots, \lambda_m)$, a constrained hinge-loss energy function *f_λ is defined as*

$$f_\lambda(\mathbf{Y}, \mathbf{X}) = \sum_{j=1}^{m} \lambda_j \phi_j(\mathbf{Y}, \mathbf{X}) \,.$$

Definition 2. *A* hinge-loss Markov random field *P over random variables $\mathbf{Y}$ and conditioned on random variables $\mathbf{X}$ is a probability density defined as follows: if $\mathbf{Y}, \mathbf{X} \notin \tilde{\mathbf{D}}$, then $P(\mathbf{Y}|\mathbf{X}) = 0$; if $\mathbf{Y}, \mathbf{X} \in \tilde{\mathbf{D}}$, then*

$$P(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z(\lambda)} \exp\left[-f_\lambda(\mathbf{Y}, \mathbf{X})\right] \,,$$

where $Z(\lambda) = \int_{\mathbf{Y}} \exp\left[-f_\lambda(\mathbf{Y}, \mathbf{X})\right]$.

Thus, MPE inference is equivalent to finding the minimizer of the convex energy f_λ.

The potentials and weights can be grouped together into *templates*, which can be used to define general classes of HL-MRFs that are parameterized by the input data. Let $\mathcal{T} = (t_1, \ldots, t_s)$ denote a vector of templates with associated weights $\Lambda = (\Lambda_1, \ldots, \Lambda_s)$. We partition the potentials by their associated templates and let $\Phi_q(\mathbf{Y}, \mathbf{X}) = \sum_{j \in t_q} \phi_j(\mathbf{Y}, \mathbf{X})$ for all $t_q \in \mathcal{T}$. In the HL-MRF, the weight of the j'th hinge-loss potential is set to the weight of the template from which it was derived, i.e., $\lambda_j = \Lambda_q$, for each $j \in t_q$.

Probabilistic soft logic [6, 13] provides a natural interface to represent hinge-loss potential templates using logical rules. In particular, a logical conjunction of Boolean variables $X \wedge Y$ can be generalized to continuous variables using the hinge function $\max\{X + Y - 1, 0\}$, which is known as the *Lukasiewicz t-norm.* Disjunction $X \vee Y$ is relaxed to $\min\{X + Y, 1\}$, and negation $\neg X$ to $1 - X$. PSL allows modelers to design rules that, given data, ground out substitutions for logical terms. The groundings of a template define hinge-loss potentials that share the same weight and have the form one minus the truth value of the ground rule. We defer to Broecheler et al. [6] and Kimmig et al. [13] for details on PSL.

To further demonstrate this templating, consider the task of predicting who trusts whom in a social network. Let the network contain three individuals: A, B, and C. We can design an HL-MRF to include potentials that encode the belief that trust is transitive (which is a rule we use in our experiments). Let the variable $Y_{A,B}$ represent how much A trusts B, and similarly so for $Y_{B,C}$ and $Y_{A,C}$. Then the potential

$$\phi(\mathbf{Y}, \mathbf{X}) = [\max\{Y_{A,B} + Y_{B,C} - Y_{A,C} - 1, 0\}]^p$$

is equivalent to one minus the truth value of the Boolean formula $Y_{A,B} \wedge Y_{B,C} \rightarrow Y_{A,C}$ when $Y_{A,B}, Y_{B,C}, Y_{A,C} \in \{0,1\}$. When they are allowed to take on their full range $[0,1]$, the potential is a convex relaxation of the implication. An HL-MRF with this potential function assigns higher probability to variable states that satisfy the logical implication above, which can occur to varying degrees in the continuous domain. Given a social network with more than these three individuals, PSL can ground out possible substitutions for the roles of A, B, and C to generate potential functions for each substitution, thus defining the full, ground HL-MRF.

HL-MRFs support a few additional components useful for modeling. The constraints in Definition 1 allow the encoding of functional modeling requirements, which is useful, e.g., when variables correspond to mutually exclusive labels, and thus should sum to one. The exponent parameter p_j allows flexibility in the shape of the hinge, affecting the sharpness of the penalty for violating the logical implication. Setting p_j to 1 penalizes violation linearly with the amount the implication is unsatisfied, while setting p_j to 2 penalizes small violations much less. In effect, some linear potentials overrule others, while the influences of squared potentials are averaged together.

3 MPE INFERENCE

MPE inference for HL-MRFs requires finding a feasible assignment that minimizes f_λ. Performing MPE inference quickly is crucial, especially because weight learning often requires performing inference many times with different weights (as we discuss in Section 4). Here, HL-MRFs have a distinct advantage over general discrete models, since minimizing f_λ is a convex optimization rather than a combinatorial one. In this section, we detail a new, faster MPE inference algorithm for HL-MRFs.

Bach et al. [3] showed how to minimize f_λ with a consensus-optimization algorithm, based on the *alternating-direction method of multipliers* (ADMM) [5]. The algorithm works by creating local copies of the variables in each potential and constraint, constraining them to be equal to the original variables, and relaxing those equality constraints to make independent subproblems. By solving the subproblems repeatedly and averaging the results, the algorithm reaches a consensus on the best values of the original variables, also called the *consensus variables*. This procedure is guaranteed to converge to the global minimizer of f_λ. See [3] and [5] for more details on consensus optimization and ADMM.

This previous consensus-optimization approach to MPE inference works well for linear potentials with at most two unobserved variables, and empirical evidence suggests it scales linearly with the size of the problem [3]. However, it is restricted to pairwise potentials and constraints, and requires an interior-point method as a subroutine to solve subproblems induced by squared potentials. Because of the embedded interior-point method, its running time can increase roughly 100-fold with squared potentials [3].

We improve the algorithm of Bach et al. [3] by reformulating the optimization to enforce $\mathbf{Y} \in [0,1]^n$ only on the consensus variables, not the local copies. This form of consensus optimization is described in greater detail by Boyd et al. [5]. The result is that, in our algorithm, the potentials and constraints are not restricted to a certain number of unknowns, and the subproblems can all be solved quickly using simple linear algebra.

Algorithm 1 gives pseudocode for our new algorithm. It starts by initializing local copies of the variables that appear in each potential and constraint, along with a corresponding Lagrange multiplier for each copy. Then, until convergence, it iteratively updates Lagrange multipliers and solves suproblems induced by the HL-MRF's potentials and constraints. If the subproblem is induced by a potential, it sets the local variable copies to a balance between the minimizer of the potential and the emerging consensus. Eventually the Lagrange multipliers will enforce agreement between the local copies and the consensus. If instead the subproblem is induced by a constraint in the HL-MRF, the algorithm projects the consensus variables' current values to that constraint's feasible region. The consensus variables are updated at each iteration based on the values of their local copies and corresponding Lagrange multipliers, and clipped to [0,1].

We take the same basic approach as Bach et al. [3] to solve the subproblems. We first try to find a minimizer on either side of the hinge (where $\ell_j(\mathbf{Y}, \mathbf{X})$ is either positive or negative) before projecting onto the plane defined by $\ell_j(\mathbf{Y}, \mathbf{X}) = 0$. Without the constraints on

Algorithm 1 MPE Inference for HL-MRFs

```
Input: HL-MRF($\mathbf{Y}, \mathbf{X}, \phi, \lambda, C, \mathcal{E}, \mathcal{I}$), $\rho > 0$
Initialize $\mathbf{y}_j$ as copies of the variables $\mathbf{Y}_j$ that appear
    in $\phi_j$, $j = 1, \dots, m$
Initialize $\mathbf{y}_{k+m}$ as copies of the variables $\mathbf{Y}_{k+m}$
    that appear in $C_k$, $k = 1, \dots, r$
Initialize Lagrange multipliers $\boldsymbol{\alpha}_i$ corresponding to
    variable copies $\mathbf{y}_i$, $i = 1, \dots, m + r$
while not converged do
  for $j = 1, \dots, m$ do
    $\boldsymbol{\alpha}_j \leftarrow \boldsymbol{\alpha}_j + \rho(\mathbf{y}_j - \mathbf{Y}_j)$
    $\mathbf{y}_j \leftarrow \mathbf{Y}_j - \boldsymbol{\alpha}_j / \rho$
    if $\ell_j(\mathbf{y}_j, \mathbf{X}) > 0$ then
      $\mathbf{y}_j \leftarrow \arg\min_{\mathbf{y}_j} \begin{bmatrix} \lambda_j [\ell_j(\mathbf{y}_j, \mathbf{X})]^{p_j} \\ +\frac{\rho}{2} \|\mathbf{y}_j - \mathbf{Y}_j + \frac{1}{\rho}\boldsymbol{\alpha}_j\|_2^2 \end{bmatrix}$
      if $\ell_j(\mathbf{y}_j, \mathbf{X}) < 0$ then
        $\mathbf{y}_j \leftarrow \mathrm{Proj}_{\ell_j = 0}(\mathbf{Y}_j)$
      end if
    end if
  end for
  for $k = 1, \dots, r$ do
    $\boldsymbol{\alpha}_{k+m} \leftarrow \boldsymbol{\alpha}_{k+m} + \rho(\mathbf{y}_{k+m} - \mathbf{Y}_{k+m})$
    $\mathbf{y}_{k+m} \leftarrow \mathrm{Proj}_{C_k}(\mathbf{Y}_{k+m})$
  end for
  for $i = 1, \dots, n$ do
    $Y_i \leftarrow \frac{1}{|\texttt{copies}(Y_i)|} \sum_{y_c \in \texttt{copies}(Y_i)} \left( y_c + \frac{\alpha_c}{\rho} \right)$
    Clip $Y_i$ to [0,1]
  end for
end while
```

the local variable copies, finding these minimizers and projections is much simpler. Solving for the constraint subproblems is now also simpler, requiring just a projection onto a hyperplane or a halfspace.

Our algorithm retains all of the benefits of the original MPE inference algorithm while removing all restrictions on the numbers of unknowns in the potentials and constraints, making MPE inference fast with both linear and squared potentials. Another advantage of consensus optimization for MPE inference is that it is easy to warm start. Warm starting provides significant efficiency gains when inference is repeated on the same HL-MRF with small changes in the weights, as often occurs during weight learning.

4 WEIGHT LEARNING

In this section, we present three weight learning methods for HL-MRFs, each with a different objective function, two of which are new for learning HL-MRFs. The first method, introduced by Broecheler et al. [6], performs approximate maximum-likelihood estimation using MPE inference to approximate the gradient of the log-likelihood. The second method maximizes the pseudolikelihood. The third method finds a large-margin solution, preferring weights that discriminate the ground truth from other states. We describe below how to apply these learning strategies to HL-MRFs.

4.1 MAXIMUM-LIKELIHOOD ESTIMATION

The canonical approach for learning parameters Λ is to maximize the log-likelihood of training data. The partial derivative of the log-likelihood with respect to a parameter Λ_q is

$$\frac{\partial \log p(\mathbf{Y}|\mathbf{X})}{\partial \Lambda_q} = \mathbb{E}_\Lambda \left[\Phi_q(\mathbf{Y}, \mathbf{X}) \right] - \Phi_q(\mathbf{Y}, \mathbf{X}) \ ,$$

where $\mathbb{E}_\Lambda$ is the expectation under the distribution defined by Λ. The voted perceptron algorithm [7] optimizes Λ by taking steps of fixed length in the direction of the gradient, then averaging the points after all steps. Any step that is outside the feasible region is projected back before continuing. For a smoother ascent, it is often helpful to divide the q-th component of the gradient by the number of groundings $|t_q|$ of the q'th template [14], which we do in our experiments. Computing the expectation is intractable, so we use a common approximation: the values of the potential functions at the most probable setting of $\mathbf{Y}$ with the current parameters [6].

4.2 MAXIMUM-PSEUDOLIKELIHOOD ESTIMATION

Since exact maximum likelihood estimation is intractable in general, we can instead perform *maximum-pseudolikelihood estimation* (MPLE) [4], which maximizes the likelihood of each variable conditioned on all other variables, i.e.,

$$\begin{aligned} P^*(\mathbf{Y}|\mathbf{X}) &= \prod_{i=1}^{n} P^*(Y_i | \mathrm{MB}(Y_i)) \\ &= \prod_{i=1}^{n} \frac{1}{Z(\lambda, Y_i)} \exp\left[-f_\lambda^i(Y_i, \mathbf{Y}, \mathbf{X}) \right] ; \\ Z(\lambda, Y_i) &= \int_{Y_i} \exp\left[-f_\lambda^i(Y_i, \mathbf{Y}, \mathbf{X}) \right] ; \\ f_\lambda^i(Y_i, \mathbf{Y}, \mathbf{X}) &= \sum_{j : i \in \phi_j} \lambda_j \phi_j \left(\{ Y_i \cup \mathbf{Y}_{\setminus i} \}, \mathbf{X} \right) . \end{aligned}$$

Here, $i \in \phi_j$ means that Y_i is involved in ϕ_j, and $\mathrm{MB}(Y_i)$ denotes the *Markov blanket* of Y_i—that is, the set of variables that co-occur with Y_i in any potential function. The partial derivative of the log-pseudolikelihood with respect to Λ_q is

$$\frac{\partial \log P^*(Y|X)}{\partial \Lambda_q} = \sum_{i=1}^{n} \mathbb{E}_{Y_i|\mathrm{MB}} \left[\sum_{j \in t_q : i \in \phi_j} \phi_j(\mathbf{Y}, \mathbf{X}) \right] - \Phi_j(\mathbf{Y}, \mathbf{X}).$$

Computing the pseudolikelihood gradient does not require inference and takes time linear in the size of $\mathbf{Y}$. However, the integral in the above expectation does not readily admit a closed-form antiderivative, so we approximate the expectation. When a variable in unconstrained, the domain of integration is a one-dimensional interval on the real number line, so Monte Carlo integration quickly converges to an accurate estimate of the expectation.

We can also apply MPLE when the constraints are not too interdependent. For example, for linear equality constraints over disjoint groups of variables (e.g., variable sets that must sum to 1.0), we can block-sample the constrained variables by sampling uniformly from a simplex. These types of constraints are often used to represent mutual exclusivity of classification labels. We can compute accurate estimates quickly because these blocks are typically low-dimensional.

4.3 LARGE-MARGIN ESTIMATION

A different approach to learning drops the probabilistic interpretation of the model and views HL-MRF inference as a prediction function. Large-margin estimation (LME) shifts the goal of learning from producing accurate probabilistic models to instead producing accurate MPE predictions. The learning task is then to find the weights Λ that provide high-accuracy structured predictions. We describe in this section a large-margin method based on the cutting-plane approach for *structural support vector machines* (SVMs) [12].

The intuition behind large-margin structured prediction is that the ground-truth state should have energy lower than any alternate state by a large margin. In our setting, the output space is continuous, so we parameterize this margin criterion with a continuous loss function. For any valid output state $\tilde{\mathbf{Y}}$, a large-margin solution should satisfy:

$$f_\lambda(\mathbf{Y}, \mathbf{X}) \leq f_\lambda(\tilde{\mathbf{Y}}, \mathbf{X}) - L(\mathbf{Y}, \tilde{\mathbf{Y}}), \forall \tilde{\mathbf{Y}},$$

where the decomposable loss function $L(\mathbf{Y}, \tilde{\mathbf{Y}}) = \sum_i L(Y_i, \tilde{Y}_i)$ measures the disagreement between a state $\tilde{\mathbf{Y}}$ and the training label state $\mathbf{Y}$. We define L as the ℓ_1 distance. Since we do not expect all problems to be perfectly separable, we relax this constraint with a penalized slack ξ. We obtain a convex learning objective for a large-margin solution

$$\begin{aligned} \min_{\Lambda \geq 0} \quad & \frac{1}{2}||\Lambda||^2 + C\xi \\ \text{s.t.} \quad & \Lambda^\top(\Phi(\mathbf{Y}, \mathbf{X}) - \Phi(\tilde{\mathbf{Y}}, \mathbf{X})) \leq -L(\mathbf{Y}, \tilde{\mathbf{Y}}) + \xi, \forall \mathbf{Y}, \end{aligned}$$

where $\Phi(\mathbf{Y}, \mathbf{X}) = (\Phi_1(\mathbf{Y}, \mathbf{X}), \ldots, \Phi_s(\mathbf{Y}, \mathbf{X}))$. This formulation is analogous to the margin-rescaling approach by Joachims et al. [12]. Though such a structured objective is natural and intuitive, its number of constraints is the cardinality of the output space, which here is infinite. Following their approach, we optimize subject to the infinite constraint set using a *cutting-plane algorithm*: we greedily grow a set K of constraints by iteratively adding the worst-violated constrain given by a *separation oracle*, then updating Λ subject to the current constraints. The goal of the cutting-plane approach is to efficiently find the set of active constraints at the solution for the full objective, without having to enumerate the infinite inactive constraints. The worst-violated constraint is

$$\arg\min_{\tilde{\mathbf{Y}}} \Lambda^\top \Phi(\tilde{\mathbf{Y}}, \mathbf{X}) - L(\mathbf{Y}, \tilde{\mathbf{Y}}).$$

The separation oracle performs loss-augmented inference by adding additional loss-augmenting potentials to the HL-MRF. For ground truth in $\{0, 1\}$, these loss-augmenting potentials are also examples of hinge-losses, and thus adding them simply creates an augmented HL-MRF. The worst-violated constraint is then computed as standard inference on the loss-augmented HL-MRF. However, ground-truth variables in the interior $(0, 1)$ cause any distance-based loss to be concave, which require the separation oracle to solve a non-convex objective. For interior ground truth values, we use the *difference of convex functions algorithm* [1] to find a local optimum. Since the concave portion of the loss-augmented inference objective pivots around the ground truth value, the subgradients are 1 or -1, depending on whether the current value is greater than the ground truth. We simply choose an initial direction for interior labels by rounding, and flip the direction of the subgradients for variables whose solution states are not in the interval corresponding to the subgradient direction until convergence.

Given a set K of constraints, we solve the SVM objective as in the primal form $\min_{\Lambda \geq 0} \frac{1}{2}||\Lambda||^2 + C\xi$ s.t. K. We then iteratively invoke the separation oracle to find the worst-violated constraint. If this new constraint is not violated, or its violation is within numerical tolerance, we have found the max-margin solution. Otherwise, we add the new constraint to K, and repeat.

One fact of note is that the large-margin criterion always requires a little slack for squared HL-MRFs. Since the squared hinge potential is quadratic and the loss is linear, there always exists a small enough distance from the ground truth such that an absolute (i.e., linear) distance is greater than the squared distance. In these cases, the slack parameter trades off between the peakedness of the learned quadratic energy function and the margin criterion.

5 EXPERIMENTS

To demonstrate the flexibility and effectiveness of HL-MRFs, we test them on four diverse learning tasks: collective classification, social-trust prediction, preference prediction, and image reconstruction. [1] Each of these experiments represents a problem domain that is best solved with relational learning approaches because structure is a critical component of their problems. The experiments show that HL-MRFs perform as well as or better than state-of-the-art approaches.

For these diverse tasks, we compare against a number of competing methods. For collective classification and social-trust prediction, we compare HL-MRFs to discrete Markov random fields (MRFs). We construct them with Markov logic networks (MLNs) [19], which template discrete MRFs using logical rules similarly to PSL for HL-MRFs. We perform inference in discrete MRFs using 2500 rounds of the sampling algorithm *MC-Sat* (500 of which are burn in), and we find approximate MPE states during MLE learning using the search algorithm *MaxWalkSat* [19]. For collaborative filtering, a task that is inherently continuous and nontrivial to encode in discrete logic, we compare against *Bayesian probabilistic matrix factorization* [20]. Finally, for image reconstruction, we run the same experimental setup as Poon and Domingos [18] and compare against the results they report, which include tests using *sum product networks*, *deep belief networks*, and *deep Boltzmann machines*.

When appropriate, we evaluate statistical significance using a paired t-test with rejection threshold 0.01. We omit variance statistics to save space and only report the average and statistical significance. We describe the HL-MRFs used for our experiments using the PSL rules that define them. To investigate the differences between linear and squared potentials we use both in our experiments. HL-MRF-L refers to a model with all linear potentials and HL-MRF-Q to one with all squared potentials. When training with MLE and MPLE, we use 100 steps of voted perceptron and a step size of 1.0 (unless otherwise noted), and for LME we set $C = 0.1$. We experimented with various settings, but the scores of HL-MRFs and discrete MRFs were not sensitive to changes.

[1]All code is available at `http://psl.umiacs.umd.edu`.

Table 1: Average accuracy of classification by HL-MRFs and discrete MRFs. Scores statistically equivalent to the best scoring method are typed in bold.

	Citeseer	Cora
HL-MRF-Q (MLE)	**0.729**	**0.816**
HL-MRF-Q (MPLE)	**0.729**	**0.818**
HL-MRF-Q (LME)	0.683	0.789
HL-MRF-L (MLE)	**0.724**	0.802
HL-MRF-L (MPLE)	**0.729**	**0.808**
HL-MRF-L (LME)	0.695	0.789
MRF (MLE)	0.686	0.756
MRF (MPLE)	0.715	0.797
MRF (LME)	0.687	0.783

5.1 COLLECTIVE CLASSIFICATION

When classifying documents, links between those documents—such as hyperlinks, citations, or co-authorship—provide extra signal beyond the local features of individual documents. Collectively predicting document classes with these links tends to improve accuracy [21]. We classify documents in citation networks using data from the Cora and Citeseer scientific paper repositories. The Cora data set contains 2,708 papers in seven categories, and 5,429 directed citation links. The Citeseer data set contains 3,312 papers in six categories, and 4,591 directed citation links.

The prediction task is, given a set of seed documents whose labels are observed, to infer the remaining document classes by propagating the seed information through the network. For each of 20 runs, we split the data sets 50/50 into training and testing partitions, and seed half of each set. To predict discrete categories with HL-MRFs we predict the category with the highest predicted value.

We compare HL-MRFs to discrete MRFs on this task. We construct both using the same logical rules, which simply encode the tendency for a class to propagate across citations. For each category C_i, we have two separate rules for each direction of citation:

$$\textsc{Label}(A, C_i) \wedge \textsc{Cites}(A, B) \Rightarrow \textsc{Label}(B, C_i),$$
$$\textsc{Label}(A, C_i) \wedge \textsc{Cites}(B, A) \Rightarrow \textsc{Label}(B, C_i).$$

Table 1 lists the results of this experiment. HL-MRFs are the most accurate predictors on both data sets. We also note that both variants of HL-MRFs are much faster than discrete MRFs. See Table 3 for average inference times on five folds.

Table 2: Average area under ROC and precision-recall curves of social-trust prediction by HL-MRFs and discrete MRFs. Scores statistically equivalent to the best scoring method by metric are typed in bold.

	ROC	P-R (+)	P-R (-)
HL-MRF-Q (MLE)	**0.822**	**0.978**	**0.452**
HL-MRF-Q (MPLE)	**0.832**	**0.979**	**0.482**
HL-MRF-Q (LME)	**0.814**	**0.976**	**0.462**
HL-MRF-L (MLE)	0.765	0.965	0.357
HL-MRF-L (MPLE)	0.757	0.963	0.333
HL-MRF-L (LME)	0.783	0.967	**0.453**
MRF (MLE)	0.655	0.942	0.270
MRF (MPLE)	0.725	0.963	0.298
MRF (LME)	0.795	**0.973**	**0.441**

5.2 SOCIAL-TRUST PREDICTION

An emerging problem in the analysis of online social networks is the task of inferring the level of trust between individuals. Predicting the strength of trust relationships can provide useful information for viral marketing, recommendation engines, and internet security. HL-MRFs with linear potentials have recently been applied by Huang et al. [10] to this task, showing superior results with models based on sociological theory. We reproduce their experimental setup using their sample of the signed Epinions trust network, in which users indicate whether they trust or distrust other users. We perform eight-fold cross-validation. In each fold, the prediction algorithm observes the entire unsigned social network and all but 1/8 of the trust ratings. We measure prediction accuracy on the held-out 1/8. The sampled network contains 2,000 users, with 8,675 signed links. Of these links, 7,974 are positive and only 701 are negative.

We use a model based on the social theory of *structural balance*, which suggests that social structures are governed by a system that prefers triangles that are considered balanced. Balanced triangles have an odd number of positive trust relationships; thus, considering all possible directions of links that form a triad of users, there are sixteen logical implications of the form

$$\textsc{Trusts}(A, B) \land \textsc{Trusts}(B, C) \Rightarrow \textsc{Trusts}(A, C).$$

Huang et al. [10] list all sixteen of these rules, a reciprocity rule, and a prior in their *Balance-Recip* model, which we omit to save space.

Since we expect some of these structural implications to be more or less accurate, learning weights for these rules provides better models. Again, we use these rules to define HL-MRFs and discrete MRFs, and we train them using various learning algorithms. We compute three metrics: the area under the receiver operating characteristic (ROC) curve, and the areas under the precision-recall curves for positive trust and negative trust. On all three metrics, HL-MRFs with squared potentials score significantly higher. The differences among the learning methods for squared HL-MRFs are insignificant, but the differences among the models is statistically significant for the ROC metric. For area under the precision-recall curve for positive trust, discrete MRFs trained with LME are statistically tied with the best score, and both HL-MRF-L and discrete MRFs trained with LME are statistically tied with the best area under the precision-recall curve for negative trust. The results are listed in Table 2.

Table 3: Average inference times (reported in seconds) of single-threaded HL-MRFs and discrete MRFs.

	Citeseer	Cora	Epinions
HL-MRF-Q	0.42	0.70	0.32
HL-MRF-L	0.46	0.50	0.28
MRF	110.96	184.32	212.36

Though the random fold splits are not the same, using the same experimental setup, Huang et al. [10] also scored the precision-recall area for negative trust of standard trust prediction algorithms EigenTrust and TidalTrust, which scored 0.131 and 0.130, respectively. The logical models based on structural balance that we run here are significantly more accurate, and HL-MRFs more than discrete MRFs.

Table 3 lists average inference times on five folds of three prediction tasks: Cora, Citeseer, and Epinions. We implemented each method in Java. Both HL-MRF-Q and HL-MRF-L are much faster than discrete MRFs. This illustrates an important difference between performing structured prediction via convex inference versus sampling in a discrete prediction space: using our MPE inference algorithm is much faster.

5.3 PREFERENCE PREDICTION

Preference prediction is the task of inferring user attitudes (often quantified by ratings) toward a set of items. This problem is naturally structured, since a user's preferences are often interdependent, as are an item's ratings. *Collaborative filtering* is the task of predicting unknown ratings using only a subset of observed ratings. Methods for this task range from simple nearest-neighbor classifiers to complex latent factor models. To illustrate the versatility of HL-MRFs, we design a simple, interpretable collaborative filtering

Table 4: Normalized mean squared/absolute errors (NMSE/NMAE) for preference prediction using the Jester dataset. The lowest errors are typed in bold.

	NMSE	NMAE
HL-MRF-Q (MLE)	0.0554	0.1974
HL-MRF-Q (MPLE)	0.0549	0.1953
HL-MRF-Q (LME)	0.0738	0.2297
HL-MRF-L (MLE)	0.0578	0.2021
HL-MRF-L (MPLE)	0.0535	0.1885
HL-MRF-L (LME)	0.0544	**0.1875**
BPMF	**0.0501**	**0.1832**

model for predicting humor preferences. We test this model on the Jester dataset, a repository of ratings from 24,983 users on a set of 100 jokes [9]. Each joke is rated on a scale of $[-10, +10]$, which we normalize to $[0, 1]$. We sample a random 2,000 users from the set of those who rated all 100 jokes, which we then split into 1,000 train and 1,000 test users. From each train and test matrix, we sample a random 50% to use as the observed features $\mathbf{X}$; the remaining ratings are treated as the variables $\mathbf{Y}$.

Our HL-MRF model uses an item-item similarity rule:

$$\textsc{SimRate}(J_1, J_2) \wedge \textsc{Likes}(U, J_1) \Rightarrow \textsc{Likes}(U, J_2)$$

where J_1, J_2 are jokes and U is a user; the predicate Likes indicates the degree of preference (i.e., rating value); and SimRate measures the mean-adjusted cosine similarity between the observed ratings of two jokes. We also include rules to enforce that $\textsc{Likes}(U, J)$ concentrates around the observed average rating of user U and item J, and the global average.

We compare our HL-MRF model to a current state-of-the-art latent factors model, *Bayesian probabilistic matrix factorization* (BPMF) [20]. BPMF is a fully Bayesian treatment and, as such, is considered “parameter-free”; the only parameter that must be specified is the rank of the decomposition.For our experiments, we use Xiong et al.’s code [2010]. Since BPMF does not train a model, we allow BPMF to use all of the training matrix during the prediction phase.

Table 4 lists the normalized mean squared error (NMSE) and normalized mean absolute error (NMAE), averaged over 10 random splits. Though BPMF produces the best scores, the improvement over HL-MRF-L (LME) is not significant in NMAE.

5.4 IMAGE RECONSTRUCTION

Digital image reconstruction requires models that understand how pixels relate to each other, such that when some pixels are unobserved, the model can infer their values from parts of the image that are observed. We construct pixel-grid HL-MRFs for image reconstruction. We test these models using the experimental setup of Poon and Domingos [18]: we reconstruct images from the Olivetti face data set and the Caltech101 face category. The Olivetti data set contains 400 images, 64 pixels wide and tall, and the Caltech101 face category contains 435 examples of faces, which we crop to the center 64 by 64 patch, as was done by Poon and Domingos [18]. Following their experimental setup, we hold out the last fifty images and predict either the left half of the image or the bottom half.

The HL-MRFs in this experiment are much more complex than the ones in our other experiments because we allow each pixel to have its own weight for the following rules, which encode agreement or disagreement between neighboring pixels:

$$\begin{aligned}
\textsc{Bright}(P_{ij}, I) \wedge \textsc{North}(P_{ij}, Q) &\Rightarrow \textsc{Bright}(Q, I),\\
\textsc{Bright}(P_{ij}, I) \wedge \textsc{North}(P_{ij}, Q) &\Rightarrow \neg\textsc{Bright}(Q, I),\\
\neg\textsc{Bright}(P_{ij}, I) \wedge \textsc{North}(P_{ij}, Q) &\Rightarrow \textsc{Bright}(Q, I),\\
\neg\textsc{Bright}(P_{ij}, I) \wedge \textsc{North}(P_{ij}, Q) &\Rightarrow \neg\textsc{Bright}(Q, I),
\end{aligned}$$

where $\textsc{Bright}(P_{ij}, I)$ is the normalized brightness of pixel P_{ij} in image I, and $\textsc{North}(P_{ij}, Q)$ indicates that Q is the north neighbor of P_{ij}. We similarly include analogous rules for the south, east, and west neighbors, as well as the pixels mirrored across the horizontal and vertical axes. This setup results in up to 24 rules per pixel, which, in a 64 by 64 image, produces 80,896 weighted potential templates.

We train these HL-MRFs using MPE-approximate maximum likelihood with a 5.0 step size on the first 200 images of each data set and test on the last fifty. For training, we maximize the data log-likelihood of uniformly random held-out pixels for each training image, allowing for generalization throughout the image. Table 5 lists our results and others reported by Poon and Domingos [18]. HL-MRFs produce the best mean squared error on the left- and bottom-half settings for the Caltech101 set and the left-half setting in the Olivetti set. Only sum product networks produce lower error on the Olivetti bottom-half faces. Some reconstructed faces are displayed in Figure 1, where the shallow, pixel-based HL-MRFs produce comparably convincing images to sum-product networks, especially in the left-half setting, where HL-MRF can learn which pixels are likely to mimic their horizontal mirror. While neither method is particularly good at

Table 5: Mean squared errors per pixel for image reconstruction. HL-MRFs produce the most accurate reconstructions on the Caltech101 and the left-half Olivetti faces, and only sum-product networks produce better reconstructions on Olivetti bottom-half faces. Scores for other methods are taken from Poon and Domingos [18].

	HL-MRF-Q (MLE)	SPN	DBM	DBN	PCA	NN
Caltech-Left	1741	1815	2998	4960	2851	2327
Caltech-Bottom	1910	1924	2656	3447	1944	2575
Olivetti-Left	927	942	1866	2386	1076	1527
Olivetti-Bottom	1226	918	2401	1931	1265	1793

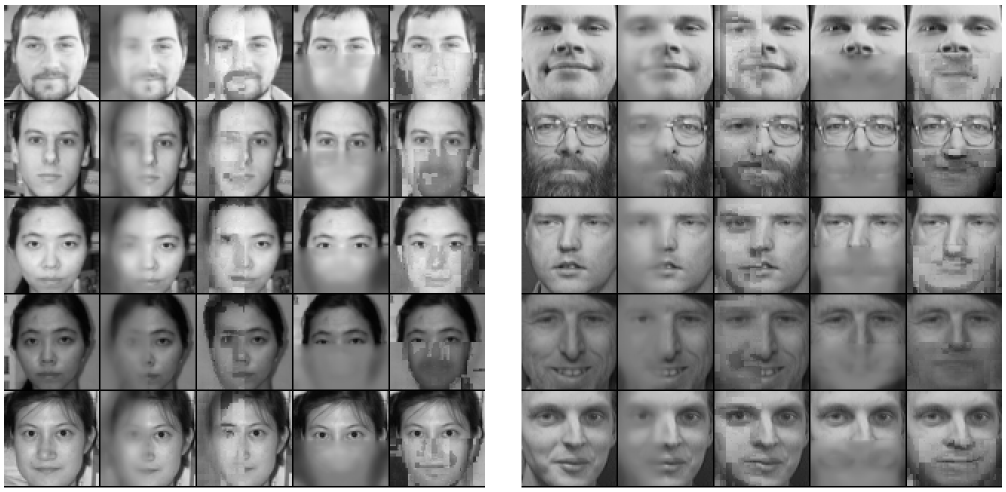


Figure 1: Example results on image reconstruction of Caltech101 (left) and Olivetti (right) faces. From left to right in each column: (1) true face, left side predictions by (2) HL-MRFs and (3) SPNs, and bottom half predictions by (4) HL-MRFs and (5) SPNs. SPN reconstructions are downloaded from Poon and Domingos [18].

reconstructing the bottom half of faces, the qualitative difference between the deep SPN and the shallow HL-MRF reconstructions is that SPNs seem to hallucinate different faces, often with some artifacts, while HL-MRFs predict blurry shapes roughly the same pixel intensity as the observed, top half of the face. The tendency to better match pixel intensity helps HL-MRFs score better quantitatively on the Caltech101 faces, where the lighting conditions are more varied.

Training and predicting with these HL-MRFs takes little time. In our experiments, training each model takes about 45 minutes on a 12-core machine, while predicting takes under a second per image. While Poon and Domingos [18] report faster training with SPNs, both HL-MRFs and SPNs clearly belong to a class of faster models when compared to DBNs and DBMs, which can take days to train on modern hardware.

6 CONCLUSION

We have shown that HL-MRFs are a flexible and interpretable class of models, capable of modeling a wide variety of domains. HL-MRFs admit fast, convex inference. The MPE inference algorithm we introduce is applicable to the full class of HL-MRFs. With this fast, general algorithm, we are the first to show results using quadratic HL-MRFs on real-world data. In our experiments, HL-MRFs match or exceed the predictive performance of state-of-the-art methods on four diverse tasks. The natural mapping between hinge-loss potentials and logic rules makes HL-MRFs easy to define and interpret.

Acknowledgements

This work was supported by NSF grants CCF0937094 and IIS1218488, and IARPA via DoI/NBC contract number D12PC00337. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

[1] L. An and P. Tao. The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Annals of Operations Research*, 133: 23–46, 2005.

[2] F. Bach and M. Jordan. Thin junction trees. In *Neural Information Processing Systems*, 2001.

[3] S. Bach, M. Broecheler, L. Getoor, and D. O'Leary. Scaling MPE inference for constrained continuous Markov random fields with consensus optimization. In *Neural Information Processing Systems*, 2012.

[4] J. Besag. Statistical analysis of non-lattice data. *Journal of the Royal Statistical Society*, 24(3): 179–195, 1975.

[5] 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), 2011.

[6] M. Broecheler, L. Mihalkova, and L. Getoor. Probabilistic similarity logic. In *Uncertainty in Artificial Intelligence*, 2010.

[7] M. Collins. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Empirical Methods in Natural Language Processing*, 2002.

[8] P. Domingos and W. Webb. A tractable first-order probabilistic logic. In *AAAI Conference on Artificial Intelligence*, 2012.

[9] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, July 2001.

[10] B. Huang, A. Kimmig, L. Getoor, and J. Golbeck. A flexible framework for probabilistic models of social trust. In *Conference on Social Computing, Behavioral-Cultural Modeling, & Prediction*, 2013.

[11] T. Huynh and R. Mooney. Online max-margin weight learning for Markov logic networks. In *SIAM International Conference on Data Mining*, 2011.

[12] T. Joachims, T. Finley, and C. Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

[13] A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor. A short introduction to probabilistic soft logic. In *NIPS Workshop on Probabilistic Programming: Foundations and Applications*, 2012.

[14] D. Lowd and P. Domingos. Efficient weight learning for Markov logic networks. In *Principles and Practice of Knowledge Discovery in Databases*, 2007.

[15] A. Martins, M. Figueiredo, P. Aguiar, N. Smith, and E. Xing. An augmented Lagrangian approach to constrained MAP inference. In *International Conference on Machine Learning*, 2011.

[16] O. Meshi and A. Globerson. An alternating direction method for dual MAP LP relaxation. In *European Conference on Machine Learning and Knowledge Discovery in Databases*, 2011.

[17] J. Neville and D. Jensen. Relational dependency networks. *Journal of Machine Learning Research*, 8:653–692, 2007.

[18] H. Poon and P. Domingos. Sum-product networks: A new deep architecture. In *Uncertainty in Artificial Intelligence*, 2011.

[19] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.

[20] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *International Conference on Machine Learning*, 2008.

[21] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[22] D. Sontag and T. Jaakkola. Tree block coordinate descent for MAP in graphical models. In *Artificial Intelligence and Statistics*, 2009.

[23] B. Taskar, M. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *Neural Information Processing Systems*, 2003.

[24] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Neural Information Processing Systems*, 2004.

[25] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2), January 2008.

[26] L. Xiong, X. Chen, T. Huang, J. Schneider, and J. Carbonell. Temporal collaborative filtering with Bayesian probabilistic tensor factorization. In *SIAM International Conference on Data Mining*, 2010.

High-dimensional Joint Sparsity Random Effects Model for Multi-task Learning

Krishnakumar Balasubramanian
Georgia Institute of Technology
krishnakumar3@gatech.edu

Kai Yu
Baidu Inc.
yukai@baidu.com

Tong Zhang
Rutgers University
tzhang@stat.rutgers.edu

Abstract

Joint sparsity regularization in multi-task learning has attracted much attention in recent years. The traditional convex formulation employs the group Lasso relaxation to achieve joint sparsity across tasks. Although this approach leads to a simple convex formulation, it suffers from several issues due to the looseness of the relaxation. To remedy this problem, we view jointly sparse multi-task learning as a specialized random effects model, and derive a convex relaxation approach that involves two steps. The first step learns the covariance matrix of the coefficients using a convex formulation which we refer to as sparse covariance coding; the second step solves a ridge regression problem with a sparse quadratic regularizer based on the covariance matrix obtained in the first step. It is shown that this approach produces an asymptotically optimal quadratic regularizer in the multitask learning setting when the number of tasks approaches infinity. Experimental results demonstrate that the convex formulation obtained via the proposed model significantly outperforms group Lasso (and related multi-stage formulations).

1 Introduction

Modern high-dimensional data sets, typically with more parameters to estimate than the number of samples available, have triggered a flurry of research based on structured sparse models, both on the statistical and computational aspects. The initial problem considered in this setting was to estimate a sparse vector under a linear model (or the Lasso problem). Recently, several approaches have been proposed for estimating a sparse vector under additional constraints, for e.g., group sparsity- where certain groups of coefficients are jointly zero or non-zero. Another closely related problem is that of multi-task learning or simultaneous sparse approximation, which are special cases of the group sparse formulation. A de-facto procedure for dealing with joint sparsity regularization is the group-Lasso estimator [16], which is based on a $(2, 1)$-mixed norm convex relaxation to the non-convex $(2, 0)$-mixed norm formulation.

However, as we shall argue in this paper, group-Lasso suffers from several drawbacks due to the looseness of the relaxation; cf., [12, 9]. We propose a general method for multi-task learning in high-dimensions based on a joint sparsity random effects model. The standard approach for dealing with random effects requires estimating covariance information. Similarly, our estimation procedure involves two-steps: a convex covariance estimation step followed by the standard ridge-regression. The first step corresponds to estimating the covariance of the coefficients under additional constraints that promote sparsity. The intuition is that to deal with group sparsity (even if we are interested in estimating the coefficients) it is better to first estimate covariance information, and then plug in the covariance estimate for estimating the coefficients. With a particular sparse diagonal structure for the covariance matrix the model becomes similar to group-lasso, and the advantage of the proposed estimation approach over group-lasso formulation will be clarified in this setting.

Related work: Traditional estimation approaches for random effects model involve two-steps: first estimate the underlying covariance matrix, and then estimate the coefficients based on the covariance matrix. However, the traditional covariance estimation procedures are non-convex such as the popular method of restricted maximum likelihood (*REML*) and such models are typically studied in the low-dimensional setting [10].

From a Bayesian perspective, a hierarchical model for simultaneous sparse approximation is proposed in [15] based on a straightforward extension of automatic relevance determination. Under that setting, the tasks share a common hyper-prior that is estimated from the data by integrating out the actual parameter. The resulting marginal likelihood is maximized for the hyper-prior parameters; this procedure is called as type-II maximum likelihood in the literature. The non-Bayesian counterpart is called *random effects model* in classical statistics, and the resulting estimator is referred to as REML. The disadvantage of this approach is that it makes the resulting optimization problem non-convex and difficult to solve efficiently, as mentioned before. In addition, the problem becomes harder to analyze and provide convincing statistical and computational guarantees, while Lasso-related formulations are well studied and favorable statistical and computational properties could be established.

More recently, the problem of joint sparsity regularization has been studied under various settings (multi-task learning [2, 1], group lasso [16], and simultaneous sparse approximation [14, 15]) in the past years. In [1], the authors develop a convex framework for multi-task learning based on the $(2,1)$-mixed norm formulation. Conditions for sparsity oracle inequalities and variable selection properties for a similar formulation are derived in [13], showing the advantage of joint estimation of tasks that share common support is statistically efficient. But the formulation has several drawbacks due to the looseness of its convex relaxation [12, 9]. The issue of bias that is inherent in the group lasso formulation was discussed in [12]. By defining a measure of sparsity level of the target signal under the group setting, the authors mention that the standard formulation of group lasso exhibits a bias that cannot be removed by simple reformulation of group lasso. In order to deal with this issue, recently [9] proposed the use of a non-convex regularizer and provided a numerical algorithm based on solving a sequence of convex relaxation problems. The method is based on a straightforward extension of a similar approach developed for the Lasso setting (cf., [17]), to the joint sparsity situation. Note that adaptive group-Lasso is a special case of [9]. In this paper, we propose a simple two-step procedure, to overcome the drawbacks of the standard group-Lasso relaxation. Compared to [9], the proposed approach is entirely convex and hence attains the global solution.

The current paper has two theoretical contributions. First, under a multi-task random effects model, we obtain an expected prediction error bound that relates the predictive performance to the accuracy of covariance estimation; by adapting high dimensional sparse covariance estimation procedures such as [8, 4], we can obtain consistent estimate of covariance matrix which leads to asymptotically optimal performance. Second, it is shown that under our random effects model, group Lasso in general does not accurately estimate the covariance matrix and thus is not optimal under the model considered. Experiments show that this approach provides improved performance compared to group Lasso (and the multi-stage versions) on simulated and real datasets.

2 Joint Sparsity Random Effects Model and Group Lasso

We consider joint sparsity regularization problems under multi-task learning. In multi-task learning, we consider m linear regression problems tasks $\ell = 1, \ldots, m$

$$y^{(\ell)} = X^{(\ell)}\bar{\beta}^{(\ell)} + \epsilon^{(\ell)}. \tag{1}$$

We assume that each $y^{(\ell)}$ is an $n^{(\ell)}$ dimensional vector, each $X^{(\ell)}$ is an $n^{(\ell)} \times d$ dimensional matrix, each $\bar{\beta}^{(\ell)}$ is the target coefficient vector for task ℓ in d dimension. For simplicity, we also assume that $\epsilon^{(\ell)}$ is an $n^{(\ell)}$ dimensional iid zero-mean Gaussian noise vector with variance σ^2: $\epsilon^{(\ell)} \sim N(0, \sigma^2 I_{n^{(\ell)}\times n^{(\ell)}})$.

The joint sparsity model in multi-task learning assumes that all $\bar{\beta}^{(\ell)}$ share similar supports: $\mathrm{supp}(\bar{\beta}^{(\ell)}) \subset \bar{F}$ for some common sparsity pattern $\bar{F}$, where $\mathrm{supp}(\beta) = \{j : \beta_j \neq 0\}$. The convex relaxation formulation for this model is given by group Lasso

$$\min_{\beta} \left[\sum_{\ell=1}^{m} \frac{1}{2} \left\| y^{(\ell)} - X^{(\ell)}\beta^{(\ell)} \right\|_2^2 + \lambda \sum_{j=1}^{d} \sqrt{\sum_{\ell=1}^{m} (\beta_j^{(\ell)})^2} \right], \tag{2}$$

where $\beta = \{\beta^{(\ell)}\}_{\ell=1,\ldots,m}$.

We observe that the multi-task group Lasso formulation (2) is equivalent to $\min_{\beta,\omega} F(\beta,\omega)$, where $F(\beta,\omega) =$

$$\sum_{\ell=1}^{m} \frac{1}{2\sigma^2} \left\| y^{(\ell)} - X^{(\ell)}\beta^{(\ell)} \right\|_2^2 + \sum_{j=1}^{d} \frac{1}{2\omega_j} \sum_{\ell=1}^{m} (\beta_j^{(\ell)})^2 + \frac{m}{2\sigma^2} \sum_{j=1}^{d} \omega_j \tag{3}$$

with $\lambda = \sigma\sqrt{m}$, where $\beta = \{\beta^{(\ell)}\}_{\ell=1,\ldots,m}$ and $\omega = \{\omega_j\}_{j=1,\ldots,d}$. With fixed hyper parameter ω, we note that (2) is a special case of

$$\min_{\beta} \sum_{\ell=1}^{m} \frac{1}{2\sigma^2} \left\| y^{(\ell)} - X^{(\ell)}\beta^{(\ell)} \right\|_2^2 + \frac{1}{2} \sum_{\ell=1}^{m} (\beta^{(\ell)})^\top \Omega^{-1} \beta^{(\ell)}, \tag{4}$$

where Ω is a hyper parameter covariance matrix shared among the tasks. This general method employs a common quadratic regularizer that is shared by all the tasks. The group Lasso formulation (2) assumes a specific form of diagonal covariance matrix $\Omega = \mathrm{diag}(\{\omega_j\})$.

Equation (4) suggests the following random effects model for joint sparsity regularization, where the coefficient vectors $\bar{\beta}^{(\ell)}$ are random vectors generated independently for each task ℓ; however they share the same covariance matrix $\bar{\Omega}$: $E\ \bar{\beta}^{(\ell)}\bar{\beta}^{(\ell)\top} = \bar{\Omega}$. Given the coefficient vector $\bar{\beta}$, we then generate $y^{(\ell)}$ based on (4). Note that we assume that Ω may contain zero-diagonal elements. If $\Omega_{jj} = 0$, then the corresponding $\bar{\beta}_j^{(\ell)} = 0$ for all ℓ. Therefore we call this model *joint sparsity random effects model* for multi-task learning.

3 Joint Sparsity via Covariance Estimation

Under the proposed joint sparsity random effects model, it can be shown (see Section 4) that the optimal quadratic optimizer $(\beta^{(\ell)})^\top\Omega^{-1}\beta^{(\ell)}$ in (2) is obtained at the true covariance $\Omega = \bar{\Omega}$. This observation suggests the following estimation procedure involving two steps:

- Step 1: Estimate the joint covariance matrix Ω as hyper parameter. In particular, this paper suggests the following method as discussed in Section 3.1: $\hat{\Omega} =$

$$\arg\min_{\Omega\in\mathcal{S}}\left[\frac{1}{2}\sum_{\ell=1}^{m}\left\|y^{(\ell)}y^{(\ell)\top} - X^{(\ell)}\Omega X^{(\ell)\top}\right\|_F^2 + R(\Omega)\right], \tag{5}$$

  where $\|\cdot\|_F$ denotes the matrix Frobenius norm, $\mathcal{S}$ is the set of symmetric positive semi-definite matrices, and $R(\Omega)$ is an appropriately defined regularizer function (specified in Section 3.1).

- Step 2: Compute each $\beta^{(\ell)}$ separately given the estimated $\hat{\Omega}$ using:

$$\hat{\beta}^{(\ell)} = \left(X^{(\ell)\top}X^{(\ell)} + \lambda\hat{\Omega}^{-1}\right)^{-1} X^{(\ell)\top}y^{(\ell)}, \tag{6}$$

  where $\ell = 1,\ldots,m$.

Note that the estimation method proposed in step 1 holds for a general class of covariance matrices. Meaningful estimates of the covariance matrix could be obtained even when the generative model assumption is violated. If the dimension d and sample size n per task are fixed, it can be shown relatively easily using classical asymptotic statistics that when $m \to \infty$, we can reliably estimate the true covariance $\bar{\Omega}$ using (5), i.e., $\hat{\Omega} \to \bar{\Omega}$. Therefore the method is asymptotically optimal as $m \to \infty$. On the other hand, the group Lasso formulation (3) produces sub-optimal estimate of ω_j, as we shall see in Section 4.2. We would like to point out that in cases when the matrix $\hat{\Omega}$ is not invertible (for example, as in the sparse diagonal case as we see next) we replace the inverse with pseudo-inverse. For ease of presentation, we use the inverse throughout the presentation, though it should be clear from the context.

3.1 Sparse Covariance Coding Models

In our two step procedure, the covariance estimation of step 1 is more complex compared to step 2, which involves only the solutions of ridge regression problems. As mentioned above, if we employ a full covariance estimation model, then the estimation procedure proposed in this work is asymptotically optimal when $m \to \infty$. However, since modern asymptotics are often concerned with the scenario when $d \gg n$, computing a $d \times d$ full matrix Ω becomes impossible without further structure on Ω. In this section, we assume that Ω is diagonal, which is consistent with the group Lasso model.

This section explains how to estimate Ω using our generative model, which implies that $\bar{\beta}^{(\ell)} \sim N(0,\Omega)$, and $y^{(\ell)} = X^{(\ell)}\bar{\beta}^{(\ell)} + \epsilon^{(\ell)}$ with $\epsilon^{(\ell)} \sim N(0, \sigma^2 I_{n^{(\ell)}\times n^{(\ell)}})$. Taking expectation of $y^{(\ell)}y^{(\ell)\top}$ with respect to ϵ and $\bar{\beta}^{(\ell)}$, we obtain $E_{\beta^{(\ell)},\epsilon}y^{(\ell)}y^{(\ell)\top} = X^{(\ell)}\Omega X^{(\ell)\top} + \sigma^2 I_{n^{(\ell)}\times n^{(\ell)}}$. This suggests the following estimator of Ω: $\hat{\Omega} =$

$$\arg\min_{\Omega\in\mathcal{S}}\sum_{\ell=1}^{m}\left\|y^{(\ell)}y^{(\ell)\top} - X^{(\ell)}\Omega X^{(\ell)\top} - \sigma^2 I_{n^{(\ell)}\times n^{(\ell)}}\right\|_F^2,$$

where $\|\cdot\|_F$ is the matrix Frobenius norm. This is equivalent to

$$\hat{\Omega} = \arg\min_{\Omega\in\mathcal{S}}\frac{1}{2}\sum_{\ell=1}^{m}\left\|y^{(\ell)}y^{(\ell)\top} - X^{(\ell)}\Omega X^{(\ell)\top}\right\|_F^2 + \lambda\mathrm{tr}\left(\Omega\sum_{\ell=1}^{m}X^{(\ell)\top}X^{(\ell)}\right) \tag{7}$$

with $\lambda = \sigma^2$. Similar ideas for estimating covariance by this approach appeared in [8, 5]. We may treat the last term as regularizer of Ω, and in such sense a more general form is to consider $\hat{\Omega} =$

$$\arg\min_{\Omega\in\mathcal{S}}\left[\frac{1}{2}\sum_{\ell=1}^{m}\left\|y^{(\ell)}y^{(\ell)\top} - X^{(\ell)}\Omega X^{(\ell)\top}\right\|_F^2 + R(\Omega)\right],$$

where $R(\Omega)$ is a general regularizer function of Ω. Note that the dimension d can be large, and thus special structure is needed to regularize Ω. In particular, to be consistent with group Lasso, we impose the diagonal covariance constraint $\Omega = \mathrm{diag}(\{\omega_j\})$, and then encourage sparsity as follows: $\hat{\Omega} =$

$$\arg\min_{\{\omega_j\geq 0\}}\sum_{\ell=1}^{m}\frac{1}{2}\|y^{(\ell)}y^{(\ell)\top} - X^{(\ell)}\mathrm{diag}(\{\omega_j\})X^{(\ell)\top}\|_F^2 + \lambda\sum_j\omega_j. \tag{8}$$

This formulation leads to sparse estimation of ω_j, which we call *sparse covariance coding (scc)*. Note that the above optimization problem is convex and hence the solution could be computed efficiently. This formulation is consistent with the group Lasso regularization which also assumes diagonal covariance implicitly as in (2). It should be noted that if the diagonals of $\sum_{\ell=1}^{m}X^{(\ell)\top}X^{(\ell)}$ have identical values, then up to a rescaling of λ, (8) is equivalent to (7) with Ω restricted to be a diagonal matrix. In the experiments conducted on real world data sets, there was no significant difference between the two regularization terms (see Table 4), when both formulations are restricted to diagonal Ω.

3.2 Other Covariance Coding Models

We now demonstrate the generality of the proposed approach for multi-task learning. Note that in addition to the sparse covariance coding method (8) that assumes a diagonal form of Ω plus sparsity constraint, some other structures may be explored. One method that has been suggested for covariance estimation in [4] is the following formulation:

$$\hat{\Omega} = \arg\min_{\Omega \in \mathcal{S}} \sum_{\ell=1}^{m} \|y^{(\ell)} y^{(\ell)\top} - X^{(\ell)} \Omega X^{(\ell)}\|_F^2 \\ +2\lambda \sum_k \gamma_k \sqrt{\sum_m \Omega_{k,m}^2}, \qquad (9)$$

where $\mathcal{S}$ denotes the set of symmetric positive semi-definite matrices $\mathcal{S}$. This approach selects a set of features, and then models a full covariance matrix within the selected set of features. Although the feature selection is achieved with a group Lasso penalty, unlike this work, [4] didn't study the possibility of using covariance estimation to do joint feature selection (which is the main purpose of this work), but rather studied covariance estimation as a separate problem.

The partial full covariance model in (9) has complexity in between that of the full covariance model and the sparse diagonal covariance model (sparse covariance coding) which we promote in this paper, at least for the purpose of joint feature selection. The latter has the smallest complexity, and thus more effective for high dimensional problems that tend to cause over-fitting.

Another model with complexity in between of sparse diagonal covariance and full covariance model is to model the covariance matrix Ω as the sum of a sparse diagonal component plus a low-rank component. This is similar in spirit to the more general sparse+low-rank matrix decomposition formulation recently appeared in the literature [7, 6, 11]. However since the sparse matrix is diagonal, identifiability holds trivially (as described in the appendix) and hence one could in principal, recover both the diagonal and the low-rank objects individually which preserves the advantages of the diagonal formulation and the richness of low-rank formulation. The model assumption is $\Omega = \Omega_S + \Omega_L$, where Ω_S is the diagonal matrix and Ω_L is the low-rank matrix. The estimation procedure now becomes the following optimization problem (and the rest follows) $[\hat{\Omega}_S, \hat{\Omega}_L] =$

$$\arg\min_{\Omega_S, \Omega_L} \sum_{\ell=1}^{m} \frac{1}{2} \|y^{(\ell)} y^{(\ell)\top} - X^{(\ell)} (\Omega_S + \Omega_L) X^{(\ell)^\top}\|_F^2 \\ + \lambda_1 \|\Omega_S\|_{\text{vec}(1)} + \lambda_2 \|\Omega_L\|_*,$$

subject to the condition that Ω_S is a non-negative diagonal matrix, and $\Omega_L \in \mathcal{S}$, where $\|\cdot\|_{\text{vec}(1)}$ is the element-wise $L1$ norm and $\|\cdot\|_*$ corresponds to trace-norm.

4 Theoretical Analysis

In this section we do a theoretical analysis of the proposed method. Specifically, we first derive upper and lower bounds for prediction error for the joint sparsity random effects model and show the optimality of the proposed approach. Informally, the notion of optimality considered is as follows: what is the 'optimal shared quadratic regularizer', when m and d goes to infinity and when solutions for each task can be written as individual ridge regression solutions with a shared quadratic regularizer (note that this includes group-Lasso method). Next, we demonstrate with a simple example (i.e., considering the low-dimensional setting) the drawback of the standard group-Lasso relaxation. In a way, this example also serves as a motivation for the approach proposed in this work and provides concrete intuition.

We consider a simplified analysis with $\hat{\Omega}$ replaced by $\hat{\Omega}^{(\ell)}$ in Step 2 so that $\hat{\Omega}^{(\ell)}$ does not depend on $y^{(\ell)}$:

$$\hat{\beta}^{(\ell)} = \left(X^{(\ell)\top} X^{(\ell)} + \lambda \hat{\Omega}^{(\ell)-1}\right)^{-1} X^{(\ell)\top} y^{(\ell)}. \qquad (10)$$

For example, this can be achieved by replacing Step 1 with $\hat{\Omega}^{(\ell)} =$

$$\arg\min_{\Omega \in \mathcal{S}} \left[\frac{1}{2} \sum_{k \neq \ell} \left\| y^{(k)} y^{(k)\top} - X^{(k)} \Omega X^{(k)\top} \right\|_F^2 + R(\Omega) \right]. \qquad (11)$$

Obviously when m is large, we have $\hat{\Omega}^{(\ell)} \approx \hat{\Omega}$. Therefore the analysis can be slightly modified to the original formulation, with an extra error term of $O(1/m)$ that vanishes when $m \to \infty$. Nevertheless, the independence of $\hat{\Omega}^{(\ell)}$ and $y^{(\ell)}$ simplifies the argument and makes the essence of our analysis much easier to understand.

4.1 Prediction Error

This section derives an expected prediction error bound for the coefficient vector $\hat{\beta}^{(\ell)}$ in (10) in terms of the accuracy of the covariance matrix estimation $\hat{\Omega}^{(\ell)}$. We consider the fixed design scenario, where the design matrices $X^{(\ell)}$ are fixed and $\epsilon^{(\ell)}$ and $\bar{\beta}^{(\ell)}$ are random.

Theorem 4.1. *Assume that $\lambda \geq \sigma^2$. For each task ℓ, given $\hat{\Omega}^{(\ell)}$ that is independent of $y^{(\ell)}$, the expected prediction error with $\hat{\beta}^{(\ell)}$ in (10) is bounded as*

$$\sigma^2 \lambda \omega^{(\ell)} \leq A \leq \lambda^2 \omega^{(\ell)},$$

where $A = E \ \|X^{(\ell)} \hat{\beta}^{(\ell)} - X^{(\ell)} \bar{\beta}^{(\ell)}\|_2^2 - \left\| X^{(\ell)} \bar{\Omega}^{1/2} \left(\bar{\Omega}^{1/2} \Sigma^{(\ell)} \bar{\Omega}^{1/2} + \lambda I\right)^{-1/2} \right\|_F^2$ *and the expectation is with respect to the random effects* $\bar{\beta}^{(\ell)}$ *and* *noise $\epsilon^{(\ell)}$, and $\Sigma^{(\ell)} = X^{(\ell)\top} X^{(\ell)}$, and*

$$\omega^{(\ell)} = \|X^{(\ell)} \left(\hat{\Omega}^{(\ell)}\Sigma^{(\ell)} + \lambda I\right)^{-1} (\hat{\Omega}^{(\ell)} - \bar{\Omega})(\Sigma^{(\ell)})^{1/2} \left((\Sigma^{(\ell)})^{1/2}\bar{\Omega}(\Sigma^{(\ell)})^{1/2} + \lambda I\right)^{-1/2} \|_F^2.$$

The bound shows that the prediction performance of (10) depends on the accuracy of estimating $\bar{\Omega}$. In particular, if $\hat{\Omega}^{(\ell)} = \bar{\Omega}$, then the optimal prediction error of $\left\|X^{(\ell)}\bar{\Omega}^{1/2}\left(\bar{\Omega}^{1/2}X^{(\ell)\top}X^{(\ell)}\bar{\Omega}^{1/2} + \lambda I\right)^{-1/2}\right\|_F^2$ can be achieved. A simplified upper bound is $E\ \|X^{(\ell)}\hat{\beta}^{(\ell)} - X^{(\ell)}\bar{\beta}^{(\ell)}\|_2^2 \leq \left\|X^{(\ell)}\bar{\Omega}^{1/2}\left(\bar{\Omega}^{1/2}\Sigma^{(\ell)}\bar{\Omega}^{1/2} + \lambda I\right)^{-\frac{1}{2}}\right\|_F^2 + \lambda^{-1}\|\Sigma^{(\ell)}(\hat{\Omega}^{(\ell)} - \bar{\Omega})\|_F^2$.

This means that if the covariance estimation is consistent; that is, if $\hat{\Omega}^{(\ell)}$ converges to $\bar{\Omega}$, then our method achieves the optimal prediction error $\left\|X^{(\ell)}\bar{\Omega}^{1/2}\left(\bar{\Omega}^{1/2}\Sigma^{(\ell)}\bar{\Omega}^{1/2} + \lambda I\right)^{-1/2}\right\|_F^2$ for all tasks.

The consistency of $\hat{\Omega}^{(\ell)}$ has been studied in the literature, for example by [4] under high dimensional sparsity assumptions. Such results can be immediately applied with Theorem 4.1 to obtain optimality of the proposed approach. Specifically, we consider the case of diagonal covariance matrix, where the sparsity in $\bar{\Omega}$ is defined as the number of non-zero diagonal entries, i.e., $s = |\{i : \Omega_{ii} \neq 0\}|$. Following [4], we consider the case $X^{(\ell)} = X \in \mathbb{R}^{n\times d}, \ell = 1, \ldots, m$. Let X_J denote the sub matrix of X obtained by removing the columns of X whose indices are not in the set J. We also assume that the diagonals of $X^\top X$ have identical values so that (8) is equivalent to (7) up to a scaling of λ.

Let $\rho_{\min}(A)$ and $\rho_{\max}(A)$ for a matrix A denote the smallest and largest eigenvalue of A respectively. We introduce two quantities [4] that impose certain assumptions on the matrix X.

Definition 1. *For $0 < t \leq d$, define $\rho_{\min}(t) := \inf_{\substack{J\subset\{1,\ldots,d\}\\ |J|\leq t}} \rho_{\min}(X_J^\top X_J)$.*

Definition 2. *The mutual coherence of the columns $X_t, t = 1, \ldots, d$ of X is defined as $\theta(X) := \max\{|X_{t'}^\top X_t|, t \neq s', 1 \leq t, t' \leq d\}$ and let $X_{\max}^2 := \max\{\|X_t\|_2^2, 1 \leq t \leq d\}$.*

We now state the following theorem establishing the consistency of covariance estimation (given by Eq 11) in the high-dimensional setting. The proof essentially follows the same argument for Theorem 8 in [4], by noticing the equivalence between (8) and (7), which implies consistency.

Theorem 4.2. *Assume that $\bar{\Omega}$ is diagonal, and $\theta(X) < \rho_{\min}(s)^2/4\rho_{\max}(X^\top X)s$. Assume n is fixed and the number of tasks and dimensionality $m, d \to \infty$ such that $\sqrt{s}\ln d/m \to 0$. Then the covariance estimator of (11), with appropriately chosen λ and $R(\Omega)$ defined by (8), converges to $\bar{\Omega}$:*

$$\|X(\hat{\Omega}^{(\ell)} - \bar{\Omega})X^\top\|_F^2 \to_P 0. \tag{12}$$

The following corollary, which is an immediate consequence of Theorem 4.1 and 4.2, establishes the asymptotic optimality (for prediction) of the proposed approach under the sparse diagonal matrix setting and $R(\Omega)$ defined as in (8). Similar result could be derived for other regularizers for $R(\Omega)$.

Corollary 1. *Under the assumption of Theorem 4.1 and 4.2, the two-step approach defined by (11) and (10), with $R(\Omega)$ defined by (8) is asymptotically optimal for prediction, for each task ℓ:*

$$E\ \|X\hat{\beta}^{(\ell)} - X\bar{\beta}^{(\ell)}\|_2^2 - \left\|X\bar{\Omega}^{1/2}\left(\bar{\Omega}^{1/2}X^\top X\bar{\Omega}^{1/2} + \lambda I\right)^{-1/2}\right\|_F^2 \to_P 0.$$

Note that the asymptotics considered above, reveals the advantage of *multi-task learning* under the joint sparsity assumption: with a fixed number of samples per each task, as the dimensions of the samples and *number of tasks* tend to infinity (obeying the condition given in theorem 4.2) the proposed two-step procedure is asymptotically optimal for prediction. Although for simplicity, we state the optimality result for (11) and (10), the same result holds for the two-step procedure given by (5) and (6), because $\hat{\Omega}^{(\ell)}$ of (11) and $\hat{\Omega}$ of (5) differ only by a factor of $O(1/m)$ which converges to zero under the asymptotics considered. Finally, we would like to remark that the mutual coherence assumption made in Theorem 4.2 could be relaxed to milder conditions (based on restricted eigenvalue type assumptions) - we leave it as future work.

4.2 Drawback of Group Lasso

In general, group Lasso does not lead to optimal performance due to looseness of the single step convex relaxation. [12, 9]. This section presents a simple but concrete example to illustrate the phenomenon and shows how $\bar{\Omega}$ is under-estimated in the group-Lasso formulation. Combined with the previous section, we have a complete theoretical justification of the superiority of our approach over group Lasso, which we will also demonstrate in the empirical study.

For this purpose, we only need to consider the following relatively simple illustration (in the low-dimensional setting). We consider the case when all design matrices equal identity: $X^{(\ell)} = I$ for $\ell = 1, \ldots, m$. This formulation is similar to *Normal means models*, a popular model in the statistics literature. It is instructive to consider this model because of its closed form solution. It helps in deriving useful insights that further help for a better understanding of more general cases. We are interested in the asymptotic behavior when $m \to \infty$ (with $n^{(\ell)}$ and d fixed), which simplifies the analysis, but nevertheless reveals the problems associated with the standard group Lasso formulation. Moreover, it should be mentioned that although the two-step procedure is motivated from a generative model, the analysis presented in this section does not need to assume that each $\beta^{(\ell)}$ is truly generated from such a model.

Proposition 1. *Suppose that $n^{(\ell)} = d$ and $X^{(\ell)} = I$ for $\ell = 1, \ldots, m$, and $m \to \infty$. The sparse covariance estimate corresponding to the formulation defined by (8) is consistent.*

Proof. The sparse covariance coding formulation (8) is equivalent to (with the intention of setting $\lambda = \sigma^2$): $\hat{\Omega}^{scc} = \arg\min_{\{\omega_j \geq 0\}} \sum_{\ell=1}^{m} \frac{1}{2} \left\| y^{(\ell)} y^{(\ell)\top} - \mathrm{diag}(\{\omega_j\}) \right\|_F^2 + \lambda m \sum_j \omega_j$. The closed form solution is given by $\hat{\omega}_j^{scc} = \max\left(0, m^{-1} \sum_{\ell=1}^{m} (y_j^{(\ell)})^2 - \lambda\right)$ for $j = 1, \ldots, d$. Since $m^{-1} \sum_{\ell=1}^{m} (y_j^{(\ell)})^2 \to E_{\beta^{(\ell)}} (\beta_j^{(\ell)})^2 + \sigma^2$ as $m \to \infty$, the variance $\hat{\omega}_j^{scc} \to E_{\beta^{(\ell)}} (\beta_j^{(\ell)})^2$ with $\lambda = \sigma^2$. Therefore $\hat{\omega}_j$ is consistent. □

Note that by plugging-in the estimate of variance into (6) with the same λ (with $\lambda = \sigma^2$), we obtain

$$\hat{\beta}_j^{(\ell)} = y_j^{(\ell)} \max\left(0, 1 - \frac{\lambda}{m^{-1} \sum_{\ell=1}^{m} (y_j^{(\ell)})^2}\right). \quad (13)$$

An immediate consequence of Proposition 1 is that the estimate define in (13) is asymptotically optimal for any method using a quadratic regularizer shared by all the tasks.

A similar analysis of group Lasso formulation would reveal its drawback. Consider the group Lasso formulation defined in (3). Under similar settings, the formulation can be written as $[\hat{\beta}, \hat{\omega}^{gl}] =$

$$\arg\min_{\beta,\omega} \sum_{\ell=1}^{m} \left\| y^{(\ell)} - \beta^{(\ell)} \right\|_2^2 + \lambda \sum_{j=1}^{d} \frac{1}{\omega_j} \sum_{\ell=1}^{m} (\beta_j^{(\ell)})^2 + m \sum_{j=1}^{d} \omega_j .$$

The closed form solution for the above formulation is given by $\hat{\omega}_j^{gl} = \max\left(0, \sqrt{\lambda m^{-1} \sum_{\ell=1}^{m} (y_j^{(\ell)})^2} - \lambda\right)$, for $j = 1, \ldots, d$, and the corresponding coefficient estimate is $\hat{\beta}_j^{(\ell)} = y_j^{(\ell)} \max\left(0, 1 - \frac{\sqrt{\lambda}}{\sqrt{m^{-1} \sum_{\ell=1}^{m} (y_j^{(\ell)})^2}}\right)$, for $\ell = 1, \ldots, m$ and $j = 1, \ldots, d$.

The solution for $\hat{\omega}_j^{gl}$ implies that it is not possible to pick a fixed λ such that the group Lasso formulation gives consistent estimate of ω_j. Since from (3), it is evident that group Lasso can also be regarded as a method that uses a quadratic regularizer shared by all the tasks, we know that the solution obtained for the corresponding co-efficient estimate is asymptotically sub-optimal. In fact, the covariance estimate $\hat{\omega}_j^{gl}$ is significantly smaller than the correct estimate $\hat{\omega}_j^{scc}$. This under-estimate of ω_j in group Lasso implies a corresponding under-estimate of $\beta^{(\ell)}$ obtained via group Lasso, when compared to (13). This under-estimation is the underlying theoretical reason why the proposed two-step procedure is superior to group Lasso for learning with joint sparsity. This claim is also confirmed by our empirical studies.

5 Experiments

We demonstrate the advantage of the proposed two-step procedure through (i) multi-task learning experiments on synthetic and real-world data sets and (ii) sparse covariance coding based image classification.

5.1 Multi-task learning

We first report illustrative experiments conducted on synthetic data sets with the proposed models. They are compared with the standard group-lasso formulation. The experimental set up is as follows: the number of tasks $m = 30$, $d = 256$, and $n^{\ell} = 150$. The data matrix consists of entries from standard Gaussian $N(0, 1)$. To generate the sparse co-efficients, we first generate a random Gaussian vector in d dimensions and set to zero $d - k$ of the co-efficients to account for sparsity. The cardinality of the set of non-zero coefficients is varied as $k = 50, 70, 90$ and the noise variance was 0.1. The results reported are averages over 100 random runs. We compare against standard group lasso, MSMTFL [9] (note that this is a non-convex approach, solved by sequence of convex relaxations) and another natural procedure (GLS-LS) where one uses group lasso for feature selection and with the selected features, one does least squares regression to estimate the coefficients. A precise theoretical comparison to MSMTFL procedure is left as future work.

Tables 2 shows the coefficient estimation error when the samples are such that they share 80% as common basis (and the rest 20% is selected randomly from the remaining basis) and when the samples share the same indices of non-zero coefficients (and the actual values vary for each signals). We note that in both cases, the model with diagonal covariance assumption and partial full covariance (Equation 9) outperforms the standard group lasso formulation, with the diagonal assumption performing better because of good estimates. The diagonal+low-rank formulation slightly outperforms the other models as it preserves the advantages of the diagonal model, while at the same time allows for additional modeling capability through the low-rank part, through proper selection of regularization parameters by cross-validation.

Support selection: While the above experiment sheds light on co-efficient estimation error, we performed another experiment to examine the selection properties of the proposed approach. Table 1 shows the hamming distance between selected basis and the actual basis using the different models. Note that Hamming distance is a desired metric for practical applications where exact recovery of the support set is not possible due to low signal-to-noise ratio. The indices with non-zero entry along the diagonal in the model with diagonal covariance assumption correspond to the selected basis. Similarly, indices with non-zero columns (or rows by symmetry) correspond to the selected basis in the partial full covariance model. The advantage of the diagonal assumption for joint feature selection is clearly seen from the table. This superiority in the feature selection process also explains the better performance achieved for coefficient estimation. A rigorous theoretical study of the feature selection properties is left as future work.

Correlated data: We next study the effect of correlated data set on the proposed approach. We generated correlated Gaussian random variables (corresponding to the size of the data matrix) in order to fill the matrix X for each task. The correlation co-efficient was fixed at 0.5. We worked with fully overlapped support set. Other problem parameters were retained. We compared the estimation accuracy of the proposed approach with different settings with group lasso and its variants. The results are summarized in Table 3. Note that the proposed approach performs much better than the group-Lasso based counterparts. Precisely characterizing this improvement theoretically would be interesting.

Next, the proposed approach was tested on three standard multi-task regression datasets (computer, school and sarcos datasets) and compared with the standard approach for multi-task learning: mixed $(2,1)$-norms or group lasso (2). A description of the datasets is given below:

Computer data set: This dataset consists of a survey among 180 people (corresponding to tasks). Each rated the likelihood of purchasing one of 20 different computers. The input consists 13 different computer characteristics, while the output corresponds to ratings. Following [1], we used the first 8 examples per task for training and the last 4 examples per task for testing.

School data set: This dataset is from the London Education Authority and consists of the exam scores of 15362 students from 139 schools (corresponding to tasks). The input consists 4 school-based and 3 student-based attributes, along with the year. The categorical features are replaced with binary features. We use 75% of the data set for training and the rest for testing.

Sarcos data set: The dataset[1] has 44,484 train samples and 4449 test samples. The task is to map a 21-dimensional input space (corresponding to characteristics of robotic arm) to the the output corresponding to seven torque measurement (tasks) to predict the inverse dynamics.

We report the average (accross tasks) root mean square error on the test data set in Table 4. Note that the proposed two-step approach performs better than the group lasso approach on all the data sets. The data sets correspond to cases with varied data size and number of tasks. Observe that even with a small training data (computer data set), performance of both our approach is better than the group-lasso approach.

5.2 SCC based Image Classification

In this section, we present a novel application of the proposed approach for obtaining sparse codes for gender recognition in CMU Multi-pie data set. The database contains 337 subjects (235 male and 102 female) across simultaneous variations in pose, expression, and illumination. The advantages of jointly coding the extracted local descriptors of an image with respect to a given dictionary for the purpose of classification has been highlighted in [3]. They propose a method based on mixed $(2,1)$-norm to jointly find a sparse representation of an image based on local descriptors of that image. Following a similar experimental setup, we use the proposed sparse covariance coding approach for attaining the same goal.

Each image is of size 30×40, size of patches is 8×8, and number of overlapping patches per image is 64. Local descriptors for each images are extracted in the form of overlapping patches and a dictionary is learned based on the obtained patches by sparse coding. With the learnt dictionary, the local descriptors of each image is jointly sparse coded via the diagonal covariance matrix assumption and the codes thus obtained are used or classification. This approach is compared with the group sparse coding based approach. Linear SVM is used in the final step for classification. Note that the purpose of the experiment is not learning a dictionary. Table 5 shows the test set and train set error for the classifier thus obtained. Note that the proposed sparse covariance coding based approach outperforms the group sparse coding based approach for gender classification due to its better quality estimates.

	Group sparse coding	Sparse cov. coding
Train error	$6.67 \pm 1.34\%$	$5.56 \pm 1.62\%$
Test error	$7.48 \pm 1.54\%$	$6.32 \pm 1.12\%$

Table 5: Face image classification based on gender: Test and Train set error rates for sparse covariance coding and group sparse coding (both with a fixed dictionary).

[1]`http://www.gaussianprocess.org/gpml/data/`

Method	80% shared basis			Completely shared basis		
	k=50	k=70	k=90	k=50	k=70	k=90
Standard group lasso	0.18	0.22	0.27	0.11	0.16	0.22
MSMTFL	0.15	0.18	0.20	0.07	0.08	0.17
Partial full covariance	0.17	0.20	0.23	0.07	0.11	0.16
Sparse diagonal covariance	0.13	0.16	0.20	0.05	0.09	0.14

Table 1: Support selection: Hamming distance between true non-zero indices and estimated non-zero indices by the indicated method for all signals.

Method	k=50	k=70	k=90
standard group Lasso	0.1541 ± 0.0045	0.1919 ± 0.0092	0.2404 ± 0.0124
GLS-LS	0.1498 ± 0.0032	0.1901 ± 0.0034	0.2383 ± 0.0342
Partial full covariance	0.1239 ± 0.0063	0.1542 ± 0.0131	0.1992 ± 0.0143
Sparse Diagonal covariance	0.1022 ± 0.0054	0.1393 ± 0.0088	0.1701 ± 0.0104
MSMTFL	0.1276 ± 0.0075	0.1564 ± 0.0153	0.1987 ± 0.0201
Diag+Low-rank covariance	0.1031 ± 0.0042	0.1212 ± 0.0122	0.1532 ± 0.0173
Standard group Lasso	0.1032 ± 0.0086	0.1574 ± 0.0151	0.1733 ± 0.0190
GLS-LS	0.1010 ± 0.0045	0.1532 ± 0.0134	0.1698 ± 0.0430
Partial full covariance	0.0735 ± 0.0078	0.1131 ± 0.0148	0.1576 ± 0.0201
Sparse Diagonal covariance	0.0447 ± 0.0071	0.0828 ± 0.0165	0.1184 ± 0.0198
MSMTFL	0.0643 ± 0.0093	0.0832 ± 0.0200	0.1457 ± 0.0223
Diag+low-rank Covariance	0.0452 ± 0.0084	0.0786 ± 0.0136	0.1012 ± 0.0161

Table 2: Coefficient estimation: Normalized L_2 distance between true coefficients and estimated coefficients by the indicated method. First 5 rows correspond to 80% shared basis and the last 5 rows correspond to fully shared basis.

6 Discussion and Future work

We proposed a two-step estimation procedure based on a specialized random effects model for dealing with joint sparsity regularization and demonstrated its advantage over the group-Lasso formulation. The proposed approach highlights the fact that enforcing interesting structure on covariance of the coefficients is better for obtaining joint sparsity in the coefficients. We leave a theoretical comparison to the MSMTFL procedure, precisely quantifying the statistical improvement provided by the proposed approach (note that MSMTFL being a non-convex procedure does not attain the global optimal solution [9]) as future work. Future work also includes (i) relaxing the assumptions made in the theoretical analysis, (ii) exploring more complex models like imposing group-mean structure on the parameters for additional flexibility, (iii) other additive decomposition of the covariance matrix with complementary regularizers and (iv) using locally-smoothed covariance estimates for time-varying joint sparsity.

A Identifiability of additive structure

The issue of identifiability (which is necessary subsequently for consistency and recovery guarantees) arises when we deal with additive decomposition of the covariance matrix. Here, we discuss about the conditions under which the model is identifiable, i.e., there exist an unique decomposition of the covariance matrix as the summation of the sparse diagonal matrix and low-rank matrix. We follow the discussion used in [11]. Let $\Omega = \Omega_s + \Omega_L$ denote the decomposition where Ω_s denotes the sparse diagonal matrix and Ω_L a low-rank matrix. Intuitively, identifiability holds if the sparse matrix is not low-rank (i.e., the support is sufficiently spread out) and the low-rank matrix is not too sparse (i.e., the singular vectors are away from co-ordinate axis). A formal argument is made based on the above intuition. We defined the following quantities (following [11]) below that measures the non-zero entries in any row or column of Ω_s and sparseness of the singular vectors of Ω_L:

$$\alpha = \max\{\|\text{sign}(\Omega_s)\|_{1\to1}, \|\text{sign}(\Omega_s)\|_{\infty\to\infty}\}$$

and

$$\beta = \|UU^T\|_\infty + \|VV^T\|_\infty + \|U\|_{2\to\infty}\|V\|_{2\to\infty},$$

where $U, V \in \mathbb{R}^{d\times r}$ are the left and right orthonormal singular vectors corresponding to non-zero singular values of Ω_L and $\|M\|_{p\to q} \overset{\text{def}}{=} \{\|Mv\|_q : v \in \mathcal{R}^m, \|v\|_p \leq 1\}$.

Note that, for a diagonal matrix, $\|\text{sign}(\Omega_s)\|_{1\to1} = \|\text{sign}(\Omega_s)\|_{\infty\to\infty} = 1$. It is proved in [11] that if $\alpha\beta < 1$, then the matrices are identifiable, i.e, the sparse plus low-rank decomposition is unique. Therefore we only need to

Method	k=50	k=70	k=90
Group Lasso	0.2012 ± 0.0033	0.2655 ± 0.0132	0.3252 ± 0.0323
GLS-LS	0.2090 ± 0.0098	0.2702 ± 0.0042	0.3304 ± 0.0333
Partial full covariance	0.1706 ± 0.0064	0.2376 ± 0.0224	0.2701 ± 0.0323
Sparse diagonal covariance	0.1634 ± 0.0022	0.2112 ± 0.0073	0.2601 ± 0.0231
MSMTFL	0.1786 ± 0.0023	0.2323 ± 0.0434	0.2776 ± 0.0223
Diag+Low-rank covariance	0.1531 ± 0.0042	0.2002 ± 0.0236	0.2544 ± 0.0145

Table 3: Coefficient estimation: Normalized L_2 distance between true coefficients and estimated coefficients by the indicated method with correlated input data.

Data set	Group lasso	MSMTFL	Sparse diagonal Covariances	Corr. Sparse diag (Eq.7)
Computer	1.542 ± 0.043	1.334 ± 0.031	1.223 ± 0.033	1.209 ± 0.054
School	2.202 ± 0.038	2.033 ± 0.241	1.987 ± 0.040	2.012 ± 0.073
Sarcos	9.221 ± 0.051	9.113 ± 0.145	8.983 ± 0.043	9.002 ± 0.032

Table 4: Multi-task learning: Average (across task) MSE error on the test data set.

require $\beta < 1$ for identifiability, which is a rather weak assumption, satisfied by most low-rank matrices with sufficient spread of the support.

B Proof of Theorem 4.1

For notational simplicity, we remove the superscripts (ℓ) in the following derivation (e.g., denote $X^{(\ell)}$ by X, $\hat{\beta}^{(\ell)}$ by $\hat{\beta}$ and so on). We have the following decomposition

$$
\begin{aligned}
&E\|X\hat{\beta} - X\bar{\beta}\|_2^2 \\
=&E\big\|X\big((X^\top X + \lambda\hat{\Omega}^{-1})^{-1}X^\top(X\bar{\beta}+\epsilon) - \bar{\beta}\big)\big\|_2^2 \\
=&E\big\|X\big(X^\top X + \lambda\hat{\Omega}^{-1}\big)^{-1}\lambda\hat{\Omega}^{-1}\bar{\beta}\big\|_2^2 \\
&+ E\big\|X\big(X^\top X + \lambda\hat{\Omega}^{-1}\big)^{-1}X^\top\epsilon\big\|_2^2 \\
=&\lambda^2\mathrm{tr}\big[X\big(X^\top X + \lambda\hat{\Omega}^{-1}\big)^{-1}\hat{\Omega}^{-1}\bar{\Omega}\hat{\Omega}^{-1}\big(X^\top X + \lambda\hat{\Omega}^{-1}\big)^{-1} \\
&X^\top\big] + \sigma^2\mathrm{tr}\big[X\big(X^\top X + \lambda\hat{\Omega}^{-1}\big)^{-1} \\
&X^\top X\big(X^\top X + \lambda\hat{\Omega}^{-1}\big)^{-1}X^\top\big] \\
\leq&\mathrm{tr}\lambda\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1}(\lambda\bar{\Omega} + \hat{\Omega}X^\top X\hat{\Omega}) \\
&\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
=&\lambda(A+B+C)
\end{aligned}
$$

where with $\Delta\hat{\Omega} = \hat{\Omega} - \bar{\Omega}$, we have $A = \mathrm{tr}\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1}\Delta\hat{\Omega}X^\top X\Delta\hat{\Omega}\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big]$ and $B = 2\mathrm{tr}\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1}\bar{\Omega}X^\top X\Delta\hat{\Omega}\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big]$ and $C = \mathrm{tr}\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1}(\bar{\Omega}X^\top X\bar{\Omega} + \lambda\bar{\Omega})\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big]$. We can further expand C as

$$
\begin{aligned}
C =&\mathrm{tr}\big[X\big(\bar{\Omega}X^\top X + \lambda I\big)^{-1}(\bar{\Omega}X^\top X\bar{\Omega} + \lambda\bar{\Omega}) \\
&\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
&-\mathrm{tr}\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1}\Delta\hat{\Omega}X^\top X\big(\bar{\Omega}X^\top X + \lambda I\big)^{-1} \\
&(\bar{\Omega}X^\top X\bar{\Omega} + \lambda\bar{\Omega})\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
=&\mathrm{tr}\big[X\bar{\Omega}\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] - \mathrm{tr}\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1} \\
&\Delta\hat{\Omega}X^\top X\bar{\Omega}\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
=&\mathrm{tr}\big[X\bar{\Omega}\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] - B/2.
\end{aligned}
$$

Therefore we have

$$
\begin{aligned}
&B + C - \mathrm{tr}\big[X\bar{\Omega}\big(X^\top X\bar{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
=&B/2 - \mathrm{tr}\big[X\bar{\Omega}\big(X^\top X\bar{\Omega} + \lambda I\big)^{-1}X^\top X\Delta\hat{\Omega} \\
&\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
=&B/2 - \mathrm{tr}\big[X\big(\bar{\Omega}X^\top X + \lambda I\big)^{-1}\bar{\Omega}X^\top X\Delta\hat{\Omega} \\
&\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
=&-\mathrm{tr}\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1}\Delta\hat{\Omega}X^\top X \\
&\big(\bar{\Omega}X^\top X + \lambda I\big)^{-1}\bar{\Omega}X^\top X\Delta\hat{\Omega}\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big].
\end{aligned}
$$

Putting all together, we have

$$
\begin{aligned}
&A + B + C - \mathrm{tr}\big[X\bar{\Omega}\big(X^\top X\bar{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
=&\mathrm{tr}\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1}\Delta\hat{\Omega}\big(I - X^\top X\big(\bar{\Omega}X^\top X + \lambda I\big)^{-1} \\
&\bar{\Omega}\big)X^\top X\Delta\hat{\Omega}\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big] \\
=&\lambda\mathrm{tr}\big[X\big(\hat{\Omega}X^\top X + \lambda I\big)^{-1}\Delta\hat{\Omega}\big(X^\top X\bar{\Omega} + \lambda I\big)^{-1} \\
&X^\top X\Delta\hat{\Omega}\big(X^\top X\hat{\Omega} + \lambda I\big)^{-1}X^\top\big].
\end{aligned}
$$

This proves the upper bound. Similarly, the lower bound follows from the fact that $E\,\|X\hat{\beta} - X\bar{\beta}\|_2^2 \geq \sigma^2(A+B+C)$.

References

[1] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[2] A. Argyriou., C. A. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. *NIPS*, 2008.

[3] S. Bengio, F. Pereira, Y. Singer, and D. Strelow. Group sparse coding. *NIPS*, 22, 2009.

[4] J. Bigot, R. Biscay, J. M. Loubes, and L. M. Alvarez. Group Lasso estimation of high-dimensional covariance matrices. *Journal of Machine Learning Research*, 2011.

[5] J. Bigot, R. Biscay, J. M. Loubes, and L. Muñiz-Alvarez. Nonparametric estimation of covariance functions by model selection. *EJS*, 4, 2010.

[6] E.J. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *JACM*, 2011.

[7] V. Chandrasekaran, S. Sanghavi, P.A. Parrilo, and A.S. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal of Optimization*, 2011.

[8] J. Fan, Y. Fan, and J. Lv. High dimensional covariance matrix estimation using a factor model. *Journal of Econometrics*, 147(1):186–197, 2008.

[9] Pinghua Gong, Jieping Ye, and Changshui Zhang. Multi-stage multi-task feature learning. *NIPS*, 2012.

[10] D.A. Harville. Maximum likelihood approaches to variance component estimation and to related problems. *JASA*, pages 320–338, 1977.

[11] D. Hsu, S. M. Kakade, and T. Zhang. Robust Matrix Decomposition with Outliers. *IEEE Transactions on Information Theory*, 2011.

[12] J. Huang and T. Zhang. The benefit of group sparsity. *The Annals of Statistics*, 38(4):1978–2004, 2010.

[13] K. Lounici, M. Pontil, A. B. Tsybakov, and S. van der Geer. Taking advantage of sparsity in multi-task learning. *COLT09*, 2009.

[14] J. A. Tropp. Algorithms for simultaneous sparse approximation. part II: Convex relaxation. *Signal Processing*, 86:589–602, 2006.

[15] D. P. Wipf and B. D. Rao. An empirical bayesian strategy for solving the simultaneous sparse approximation problem. *Signal Processing, IEEE Transactions on*, 55(7):3704–3716, 2007.

[16] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *JRSS: Series B*, 68, 2006.

[17] Tong Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *JMLR*, 2010.

Lower Bounds for Exact Model Counting and Applications in Probabilistic Databases*

Paul Beame **Jerry Li** **Sudeepa Roy** **Dan Suciu**
Computer Science and Engineering
University of Washington
Seattle, WA 98195
{beame, jerryzli, sudeepa, suciu}@cs.washington.edu

Abstract

The best current methods for exactly computing the number of satisfying assignments, or the satisfying probability, of Boolean formulas can be seen, either directly or indirectly, as building *decision-DNNF (decision decomposable negation normal form)* representations of the input Boolean formulas. Decision-DNNFs are a special case of *d*-DNNFs where *d* stands for *deterministic*. We show that any decision-DNNF can be converted into an equivalent *FBDD (free binary decision diagram)* – also known as a *read-once branching program (ROBP or 1-BP)* – with only a quasipolynomial increase in representation size in general, and with only a polynomial increase in size in the special case of monotone *k*-DNF formulas. Leveraging known exponential lower bounds for FBDDs, we then obtain similar exponential lower bounds for decision-DNNFs which provide lower bounds for the recent algorithms. We also separate the power of decision-DNNFs from *d*-DNNFs and a generalization of decision-DNNFs known as AND-FBDDs. Finally we show how these imply exponential lower bounds for natural problems associated with probabilistic databases.

1 Introduction

Model counting is the problem of computing the number, $\#F$, of satisfying assignments of a Boolean formula F. While model counting is hard for $\#\mathsf{P}$, there have been major advances in practical algorithms that compute exact model counts for many relatively complex formulas and, using similar techniques, that compute the probability that Boolean formulas are satisfied, given independent probabilities for their literals.

Modern exact model counting algorithms use a variety of techniques (see [Gomes et al., 2009] for a survey). Many are based on extensions of backtracking search using the *DPLL* family of algorithms [Davis and Putnam, 1960, Davis et al., 1962] that were originally designed for satisfiability search. In the context of model counting (and related problems of exact Bayesian inference) extensions include caching the results of solved sub-problems [Majercik and Littman, 1998], dynamically decomposing residual formulas into components (Relsat [Bayardo et al., 2000]) and caching their counts ([Bacchus et al., 2003]), and applying dynamic component caching together with conflict-directed clause learning (CDCL) to further prune the search (Cachet [Sang et al., 2004] and sharpSAT [Thurley, 2006]).

The other major approach, known as *knowledge compilation*, is to convert the input formula into a representation of the Boolean function that the formula defines and from which the model count can be computed efficiently in the size of the representation [Darwiche, 2001a, Darwiche, 2001b, Huang and Darwiche, 2007, Muise et al., 2012]. Efficiency for knowledge compilation depends both on the size of the representation and the time required to construct it. As noted by (`c2d` [Huang and Darwiche, 2007] based on component caching) and (Dsharp [Muise et al., 2012] based on sharpSAT), the traces of all the DPLL-based methods yield knowledge compilation algorithms that can produce what are known as *decision-DNNF* representations [Huang and Darwiche, 2005, Huang and Darwiche, 2007], a syntactic subclass of *d*-DNNF representations [Darwiche, 2001b, Darwiche and Marquis, 2002]. Indeed, all the methods for exact model counting surveyed in [Gomes et al., 2009] (and all others of which we are aware) can be converted to knowledge com-

*This work was partially supported by NSF IIS-1115188, IIS-0915054, and CCF-1217099.

pilation algorithms that produce decision-DNNF representations, without any significant increase in their running time.

In this paper we prove exponential lower bounds on the size of decision-DNNFs for natural classes of formulas. Therefore our results immediately imply exponential lower bounds for all modern exact model counting algorithms. These bounds are unconditional – they do not depend on any unproved complexity-theoretic assumptions. These bounds apply to very simple classes of Boolean formulas, which occur frequently both in uncertainty reasoning, and in probabilistic inference. We also show that our lower bounds extend to the evaluation of the properties of a large class of database queries, which have been studied in the context of probabilistic databases.

We derive our exponential lower bounds by showing how to translate any decision-DNNF to an equivalent *FBDD*, a less powerful representation for Boolean functions. Our translation increases the size by at most a quasipolynomial, and by at most a polynomial in the special case when the Boolean function computed has a monotone k-DNF formula. The lower bounds follow from well-established exponential lower bounds for FBDDs. This translation from decision-DNNFs to FBDDs is of independent interest: it is simple, and efficient, in the sense that it can be computed in time linear in the size of the output FBDD.

It is interesting to note that with formula caching, but without dynamic component caching, the trace extensions of DPLL-based searches yield FBDDs rather than decision-DNNFs. Hence, the difference between FBDDs and decision-DNNFs is precisely the ability of the latter to take advantage of decompositions into connected components of subformulas of the formula being represented. Our conversion shows that these connected component decompositions can only provide quasipolynomial improvements in efficiency, or only a polynomial improvement in the case of monotone k-DNF formulas.

Representations Though closely related, FBDDs and decision-DNNFs originate in completely different approaches for representing (or computing) Boolean functions. FBDDs are special kinds of *binary decision diagrams* [Akers, 1978], also known as *branching programs* [Masek, 1976]. These represent a function using a directed acyclic graph with *decision* nodes, each of which queries a Boolean variable representing an input bit and has 2 out-edges, one labeled 0 and the other 1; it has a single source node, and has sink nodes labeled by output values; the value of the function on an assignment of the Boolean variables is the label of the sink node reached. *Free* binary decision diagrams (FBDDs), also known as *read-once* branching programs (ROBPs), have the property that each input variable is queried at most once on each source-sink path[1]. There are many variants and extensions of these decision-based representations; for an extensive discussion of their theory see the monograph [Wegener, 2000]. These include nondeterministic extensions of FBDDs called *OR-FBDDs*, as well as their corresponding co-nondeterministic extensions called *AND-FBDDs*, which have additional internal AND nodes through which any input can pass – the output value is 1 for an input iff *every* consistent source-sink path leads to a sink labeled 1.

Decision-DNNFs originate in the desire to find restricted forms of Boolean circuits that have better properties for knowledge representation. *Negation normal form (NNF)* circuits are those that have unbounded fan-in AND and OR nodes (gates) with all negations pushed to the input level using De Morgan's laws. Darwiche [Darwiche, 2001a] introduced *decomposable negation normal form (DNNF)* which restricts NNF by requiring that the sub-circuits leading into each AND gate are defined on disjoint sets of variables. He also introduced d-DNNFs [Darwiche, 2001a, Darwiche and Marquis, 2002] which have the further restriction that DNNFs are *deterministic*, *i.e.*, the sub-circuits leading into each OR gate never simultaneously evaluate to 1; d-DNNFs have the advantage of probabilistic polynomial-time equivalence testing [Huang and Darwiche, 2007]. Most subsequent work has used these d-DNNFs. An easy way of ensuring determinism is to have a single variable x that evaluates to 1 on one branch and 0 on the other, so d-DNNFs can be produced by the subcircuit $(x \wedge A) \vee (\neg x \wedge B)$, which is equivalent to having decision nodes as above; moreover, the decomposability ensures that x does not appear in either A or B. d-DNNFs in which all OR nodes are of this form are called decision-DNNFs [Huang and Darwiche, 2005, Huang and Darwiche, 2007]. Virtually all algorithmic methods that use d-DNNFs, including those used in exact model counting and Bayesian inference, actually ensure determinism by using decision-DNNFs. Decision-DNNFs have the further advantage of being *syntactically* checkable; by comparison, given a general DNNF, it is not easy to check whether it satisfies the semantic restriction of being a d-DNNF.

[1]The term *free* contrasts with *ordered* binary decision diagrams (OBDDs) [Bryant, 1986] in which each root-leaf path must query the variables in the same order. For each variable order, minimized OBDDs are canonical representations for Boolean functions, making them extremely useful for a vast number of applications. Unfortunately, OBDDs are often also simply referred to as BDDs, which leads to confusion with the original general model.

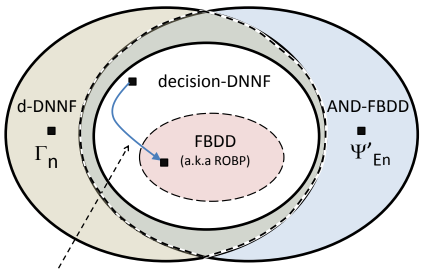

decision-DNNF
d-DNNF
Γ_n
FBDD
(a.k.a ROBP)
AND-FBDD
Ψ'_{En}
• Quasi-polynomial increase (general formula)
• Polynomial increase (monotone k-DNF)

Figure 1: A summary of our contributions (see Section 3). Here, one representation is contained in another if and only if the first can be (locally) translated into the second with at most a polynomial increase in size.

It is immediate that one can get a completely equivalent representation to the above definition by using a decision node on x in place of each OR of ANDs involving x, and in place of each leaf variable or its negation; the decomposability property ensures that no root-leaf path in the circuit queries the same variable more than once. Clearly these form a special subclass of the AND-FBDDs discussed above, in which each AND is required to have the decomposability property that the different branches below each AND node query disjoint sets of variables. Though formally there are insignificant syntactic differences between the definitions, we will use the term decision-DNNFs to refer to these *decomposable* AND-FBDDs.

Two other consequences of our simulation of decision-DNNFs by FBDDs are provable exponential separations between the representational power of decision-DNNFs and that of either d-DNNFs or AND-FBDDs. There are two functions, involving simple tests on the rows and columns of Boolean matrices, that require exponential size FBDDs but have linear size representations as AND-FBDDs and d-DNNFs respectively (cf. Thms 10.3.8, 10.4.7. in [Wegener, 2000]); our simulation shows that these lower bounds carry over to decision-DNNFs, yielding the claimed separations. A comparison of these representations in terms of their succinctness as well as a summary of our contributions in this paper are given in Figure 1[2].

Probabilistic Databases These databases annotate each tuple with a probability of being true [Suciu et al., 2011]. Query evaluation on probabilistic databases reduces to the problem of computing the probability of a positive, k-DNF Boolean formula, where the number of Boolean variables in each term is bounded by k, which is fixed by the query, while the size of the formula grows polynomially in the size of the database. Our results immediately imply that, when applied to such formulas, decision-DNNFs are only polynomially more concise than FBDDs. By combining this with previously known results, we describe a class of queries such that any query in this class generates Boolean formulas requiring decision-DNNFs of exponential size, thus implying that none of the recent evaluation algorithms that either explicitly or implicitly yield decision-DNNFs can compute these queries efficiently. Although the exponential lower bounds we derive for decision-DNNFs are not the first – there are a small number of exponential lower bounds known even for unrestricted AND-FBDDs [Wegener, 2000], which therefore also apply for decision-DNNFs – none of these apply to the kinds of simple structured properties that show up in probabilistic databases that we are able to analyze.

Compilation As noted above, the size of the decision-DNNF required is not the only source of complexity in exact model counting. The other source is the search or compilation process itself – the time required to produce a decision-DNNF from an input Boolean formula which may greatly exceed the size of the representation. A particularly striking case where this is an issue is that of an unsatisfiable Boolean formula for which the function evaluates to the constant 0 and hence the decision-DNNF is of size 1. Determining this fact may take exponential time. Indeed, DPLL with caching and conflict-directed clause learning is a special case of resolution theorem proving [Beame et al., 2004]. There are large numbers of unsatisfiable formulas for which exponential lower bounds are known for every resolution refutation (see, *e.g.*, [Ben-Sasson and Wigderson, 2001]) and hence this compilation process must be exponential for such formulas[3]. The same issues can arise in ruling out parts of the space of assignments for satisfiable formulas. However, we do not know of any lower bounds for this excess compilation time that directly apply to the kinds of simple highly satisfiable instances that we discuss in this paper.

The rest of the paper is organized as follows. In Section 2 we review FBDDs and decision-DNNFs. Section 3 presents our two main results: a general transformation of a decision-DNNF into an equivalent FBDD, with only a quasipolynomial increase in size in general, and only a polynomial increase in size for monotone k-DNF formulas. We prove these results in Section 4 and Section 5. In Section 6 we discuss the implications of this transformation for evaluating queries in probabilistic databases. We conclude in Section 7.

[2]It is open whether the region is empty if no black square is shown (also indicated by dotted borders).

[3]DPLL with formula caching, but not clause learning, can be simulated by even simpler *regular* resolution, though in general it is not quite as powerful as regular resolution [Beame et al., 2010].

2 FBDDs and Decision-DNNFs

FBDDs. An FBDD is a rooted directed acyclic graph (DAG) $\mathcal{F}$, with two kinds of nodes: *decision nodes*, each labeled by a Boolean variable X and two outgoing edges labeled 0 and 1, and *sink nodes* labeled 0 and 1. Every path from the root to some leaf node may test a Boolean variable X at most once. The size of the FBDD is the number of its nodes. We denote the sub-DAG of $\mathcal{F}$ rooted at an internal node u by $\mathcal{F}_u$ which computes a Boolean function Φ_u; $\mathcal{F}$ computes Φ_r where r is the root. For a node u labeled X with 0- and 1-children u_0 and u_1, $\Phi_u = (\neg X)\Phi_{u_0} \vee X\Phi_{u_1}$. The probability of Φ_r can be computed in linear time in the size of the FBDD using a simple dynamic program: $\Pr[\Phi_u] = (1 - p(X))\Pr[\Phi_{u_0}] + p(X)\Pr[\Phi_{u_1}]$.

Decision-DNNFs As noted in the introduction, we choose to define decision-DNNFs as a sub-class of AND-FBDDs. An AND-FBDD [Wegener, 2000] is an FBDD with an additional kind of nodes, called AND-nodes; the function associated to an AND-node u with children $u_1, \dots, u_r$ is $\Phi_u = \Phi_{u_1} \wedge \dots \wedge \Phi_{u_r}$. A decision-DNNF, $\mathcal{D}$, is an AND-FBDD satisfying the additional restriction that for any AND-node u and distinct children u_i, u_j of u, the sub-DAGS $\mathcal{D}_{u_i}$ and $\mathcal{D}_{u_j}$ do not mention any common Boolean variable X.

For the rest of the paper we make two assumptions about decision-DNNFs. First, every AND-node has exactly 2 children, and as a consequence every internal node u has exactly two children v_1, v_2, called the *left* and *right child* respectively; second, that every 1-sink node has at most one incoming edge. Both assumptions are easily enforced by at most a quadratic increase in the number of nodes in the decision-DNNF.

3 Main Results

In this section we state our two main results and show several applications. We first need some notation. For each node u of a decision-DNNF $\mathcal{D}$, let M_u be the number of AND-nodes in the subgraph $\mathcal{D}_u$. If u is an AND-node, then we have $M_u = 1 + M_{v_1} + M_{v_2}$, because, by definition, the two DAGs $\mathcal{D}_{v_1}$ and $\mathcal{D}_{v_2}$ are disjoint; we will always assume that $M_{v_1} \leq M_{v_2}$ (otherwise we swap the two children of the AND-node u), and this implies that $M_u \geq 2M_{v_1} + 1$. We classify the edges of the decision-DNNF into three categories: (u, v) is a *light edge* if u is an AND-node and v its first child; (u, v) is a *heavy edge* if u is an AND-node and v is a its second child; and (u, v) is a *neutral edge* if u is a decision node. We always have $M_u \geq M_v$, while for a light edge we have $M_u \geq 2M_v + 1$.

Let $\mathcal{D}$ be a decision-DNNF, N the total number of nodes in $\mathcal{D}$, M the number of AND-nodes, and L the maximum number of light edges on any path from the root node to some leaf node. Our first main result is:

Theorem 3.1. *For any decision-DNNF $\mathcal{D}$ there exists an equivalent FBDD $\mathcal{F}$ computing the same formula as $\mathcal{D}$, with at most NM^L nodes. Moreover, given $\mathcal{D}$, $\mathcal{F}$ can be constructed in time $O(NM^L)$.*

We give the proof in Section 4. We next show that the bound NM^L is quasipolynomial in N.

Corollary 3.2. *For any decision-DNNF $\mathcal{D}$ with N nodes there exists an equivalent FBDD $\mathcal{F}$ with at most $N2^{\log^2 N}$ nodes.*

Proof. Consider any path in $\mathcal{D}$ with L light edges, $(u_1, v_1), (u_2, v_2), \dots, (u_L, v_L)$. We have $M_{u_i} \geq 2M_{v_i} + 1$ and $M_{v_i} \geq M_{u_{i+1}}$ for all i, and we also have $M \geq M_{u_1}$ and $M_{v_L} \geq 0$, which implies $M \geq 2^L - 1$ (by induction on L). Therefore, $2^L \leq M + 1 \leq N$ (because $\mathcal{D}$ has at least one node that is not an AND-node), and $NM^L = N2^{L \log M} \leq N2^{\log^2 N}$, proving the claim. □

Our second main result concerns monotone k-DNF Boolean formulas, which have applications to probabilistic databases, as we explain in Section 6. We show that in this case any decision-DNNF can be converted into an equivalent FBDD with only a polynomial increase in size. This results from the following lemma, whose proof we give in Section 5:

Lemma 3.3. *If a decision-DNNF $\mathcal{D}$ computes a monotone k-DNF Boolean formula then every path in $\mathcal{D}$ has at most $k - 1$ AND-nodes.*

Therefore, $L \leq k - 1$, and Theorem 3.1 implies:

Theorem 3.4. *For any decision-DNNF $\mathcal{D}$ with N nodes that computes a monotone k-DNF Boolean formula then there exists an equivalent FBDD $\mathcal{F}$ with at most N^k nodes.*

We give now several applications of our main results.

Lower Bounds for DPLL-based Algorithm We give an explicit Boolean formula on which every DPLL-based algorithm whose trace is a decision-DNNF takes exponential time. We use the following formula introduced by Bollig and Wegener [Bollig and Wegener, 1998]. For any set $E \subseteq [n] \times [n]$ define $\Psi_E = \bigvee_{(i,j)\in E} X_i Y_j$, where $X_1, \dots, X_n, Y_1, \dots, Y_n$ are Boolean variables. Let $n = p^2$ where p is a prime number; then each number $0 \leq i < n$ can be uniquely written as $i = a + bp$ where $0 \leq a, b < p$. Define $E_n = \{(i + 1, j + 1) \mid i = a + bp, j = c + dp, c \equiv (a + bd) \mod p\}$. Then:

Theorem 3.5. *[Bollig and Wegener, 1998, Th.3.1] Any FBDD for Ψ_{E_n} has $2^{\Omega(\sqrt{n})}$ nodes.*

Consider the formula $\Phi_n = \bigvee_{1\leq i,j\leq n} X_i Z_{ij} Y_j$. Any FBDD for Φ_n has size $2^{\Omega(\sqrt{n})}$, because it can be converted into an FBDD for Ψ_{E_n} by setting $Z_{ij} = 1$ or

$Z_{ij} = 0$, depending on whether (i, j) is in E_n or not. Both Ψ_{E_n} and Φ_n are monotone, and 2-DNF and 3-DNF respectively, therefore, by Theorem 3.4:

Corollary 3.6. *Any decision-DNNF for either Ψ_{E_n} or Φ_n has $2^{\Omega(\sqrt{n})}$ nodes.*

In particular, any DPLL-based algorithm whose trace is a decision-DNNF will take exponential time on the formulas Ψ_{E_n} and Φ_n.

Separating decision-DNNFs from AND-FBDDs We show that decision-DNNFs are strictly weaker than AND-FBDDs. Define $\Psi'_{E_n} = \bigwedge_{(i,j)\in E_n}(X_i \vee Y_j)$, the CNF expression that is the dual of Ψ_{E_n}. Since Ψ'_{E_n} is a CNF formula, it admits an AND-FBDD with at most n^2 nodes (since $|E_n| \leq n^2$). On the other hand, we show that any decision-DNNF must have $\Omega(2^{n^{1/4}})$ nodes. Indeed, Theorem 3.5 implies that any FBDD for Ψ'_{E_n} has $2^{\Omega(\sqrt{n})}$ nodes. Consider some decision-DNNF $\mathcal{D}$ for Ψ'_{E_n} having N nodes. By Corollary 3.2 we obtain an FBDD $\mathcal{F}$ of size $2^{\log^2 N + \log N}$, which must be $2^{\Omega(\sqrt{n})}$; thus $\log^2 N = \Omega(\sqrt{n})$, hence $\log N = \Omega(n^{1/4})$, and $N = 2^{\Omega(n^{1/4})}$. We have shown:

Corollary 3.7. *Decision-DNNFs are exponentially less concise than AND-FBDDs.*

Separating decision-DNNFs from d-DNNFs Define Γ_n on the matrix of variables X_{ij} for $i, j \in [n]$ by $\Gamma_n(X) = f_n(X) \vee g_n(X)$ where f_n is 1 if and only if the parity of all the variables is even and the matrix has an all-1 row and g_n is 1 if and only if the parity of all the variables is odd and the matrix has an all-1 column. Wegener showed (cf. Theorem 10.4.7. in [Wegener, 2000]) that any FBDD for Γ_n has $2^{\Omega(n)}$ nodes (therefore, every decision-DNNF requires $2^{\Omega(n^{1/2})}$ nodes). Γ_n can also be computed by an $O(n^2)$ size d-DNNF, because both f_n and g_n can be computed by $O(n^2)$ size OBDDs, and $f_n \wedge g_n \equiv$ `false`. Hence:

Corollary 3.8. *Decision-DNNFs are exponentially less concise than d-DNNFs.*

4 Decision-DNNF to FBDD

In this section we prove Theorem 3.1 by describing a construction to convert a decision-DNNF $\mathcal{D}$ to an FBDD $\mathcal{F}$.

4.1 Main Ideas

To construct $\mathcal{F}$ we must remove all AND-nodes in $\mathcal{D}$ and replace them with decision nodes. An AND node has two children, v_1, v_2; we need to replace this node with an FBDD for the expression $\Phi_{v_1} \wedge \Phi_{v_2}$. Assume that u is the only AND-node in $\mathcal{D}$; then both $\mathcal{D}_{v_1}$ and $\mathcal{D}_{v_2}$ are already FBDDs, and Figure 2(a): stack

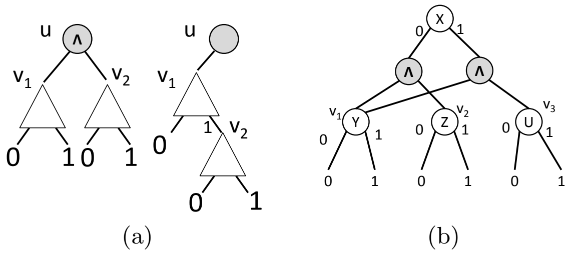


Figure 2: (a) Basic construction for converting a decision-DNNF into an FBDD (b) where it fails.

$\mathcal{D}_{v_1}$ over $\mathcal{D}_{v_2}$, and redirect all 1-sink nodes in $\mathcal{D}_{v_1}$ to the root of $\mathcal{D}_{v_2}$. Clearly this computes the same AND function; moreover it is a correct FBDD because $\mathcal{D}_{v_1}, \mathcal{D}_{v_2}$ do not have any common variable. If this construction worked in general, then the entire decision-DNNF would be converted into an FBDD of exactly the same size.

But, in general, this simple idea fails, as can be seen on the simple decision-DNNF in Figure 2(b) (it computes $(\neg X)YZ \vee XYU$). To compute the first AND node we need to stack $\mathcal{D}_{v_1}$ over $\mathcal{D}_{v_2}$, and to compute the second AND node we need to stack $\mathcal{D}_{v_1}$ over $\mathcal{D}_{v_3}$: this creates a conflict for redirecting the 1-sink node in $\mathcal{D}_{v_1}$ to v_2 or to v_3[4]. To get around that, we use two ideas. The first idea is to make copies of some subgraphs. For example if we make two copies of $\mathcal{D}_{v_1}$, call them $\mathcal{D}_{v_1}$ and $\mathcal{D}'_{v_1}$, then we can compute the first AND-node by stacking $\mathcal{D}_{v_1}$ over $\mathcal{D}_{v_2}$, and compute the second AND-node by stacking $\mathcal{D}'_{v_1}$ over $\mathcal{D}_{v_3}$ and the conflict is resolved. The second idea is to reorder the children of the AND-nodes to limit the exponential blowup due to copying. We present the details next.

4.2 The Construction of $\mathcal{F}$

Fix the decision-DNNF $\mathcal{D}$. Let u denote a node in $\mathcal{D}$ and P denote a path from the root to u. Let $s(P)$ be the set of light edges on the path P, and let $S(u)$ consist of the sets $s(P)$ for all paths from the root to u, formally:

$$s(P) = \{(v, w) \mid (v, w) \text{ is a light edge in } P\}$$
$$S(u) = \{s(P) \mid P \text{ is a path from the root to } u\}$$

We consider the light edges in a set $s = s(P)$ ordered by their occurrences in P (from the root to u). This order is independent of P: if $s = s(P) = s(P')$ then the light edges occur in the same order on the paths P and P' (since $\mathcal{D}$ is acyclic).

We will convert $\mathcal{D}$ into an FBDD $\mathcal{F}$ with *no-op nodes*, unlabeled nodes having only one outgoing edge. Any

[4]In this particular example one could stack $\mathcal{D}_{v_2}$ and $\mathcal{D}_{v_3}$ over $\mathcal{D}_{v_1}$ and avoid the conflict; but, in general, $\mathcal{D}_{v_2}, \mathcal{D}_{v_3}$ may have conflicts with other subgraphs.

FBDD with no-op nodes is easily transformed into a standard FBDD by removing the no-op nodes and redirecting all incoming edges to its unique child.

We define $\mathcal{F}$ formally. Its nodes are pairs (u, s) where u is a node in $\mathcal{D}$ and $s \in S(u)$. The root node is $(\texttt{root}(\mathcal{D}), \emptyset)$. The edges in $\mathcal{F}$ are of three types:
Type 1: For each light edge $e = (u, v)$ in $\mathcal{D}$ and every $s \in S(u)$, add the edge $((u, s), (v, s \cup \{e\}))$ to $\mathcal{F}$,
Type 2: For every neutral edge (u, v) in $\mathcal{D}$ and every $s \in S(u)$ add the edge $((u, s), (v, s))$ to $\mathcal{F}$,
Type 3: For every heavy edge (u, v_2), let $e = (u, v_1)$ be the corresponding light sibling edge. Then, for every $s \in S(u)$, add all edges of the form $((w, s \cup \{e\}), (v_2, s))$, where w is a 1-sink node in $\mathcal{D}_{v_1}$, v_2 is the heavy child of u, $s \cup \{e\} \in S(w)$, and $s \in S(v_2)$.

Finally, we label every node $u' = (u, s)$ in $\mathcal{F}$, as follows: (1) If u is a decision node in $\mathcal{D}$ that tests the variable X, then u' is a decision node in $\mathcal{F}$ testing the same variable X, (2) If u is an AND-node, then u' is a no-op node, (3) If u is a 0-sink node, then u' is a 0-sink node, (4) If u is a 1-sink node, then: if $s = \emptyset$ then u' is a 1-sink node, otherwise it is a no-op node.

This completes our description of $\mathcal{F}$. The intuition behind it is that, for every AND node, we make a fresh copy of its left child. To illustrate this, suppose $\mathcal{D}$ has a single AND-node u with two children v_1, v_2, and let $e = (u, v_1)$ be the light edge. Suppose there is a second, neutral edge into v_1, say (z, v_1). Then $\mathcal{F}$ contains two copies of the subgraph $\mathcal{D}_{v_1}$, one with nodes labeled $(w, \{e\})$, and the other with nodes labeled $(w, \emptyset)$. Any 1-sink node in the first copy becomes a no-op node in $\mathcal{F}$ and is connected to v_2, similarly to Figure 2(a); the same 1-sink node in the second copy remain 1-sink nodes. This copying process is repeated in $\mathcal{D}_{v_1}$.

4.3 Proof of Theorem 3.1

Theorem 3.1 follows from the following three lemmas:

Lemma 4.1. *$\mathcal{F}$ has at most NM^L nodes.*

Lemma 4.2. *$\mathcal{F}$ is a correct FBDD with no-op nodes.*

Lemma 4.3. *$\mathcal{F}$ computes the same function as $\mathcal{D}$.*

Proof of Lemma 4.1. The nodes of $\mathcal{F}$ have the form (u, s). There are N possible choices for the node u, and at most M^L possible choices for the set s, because $|s| \leq L$ (since every path has $\leq L$ light edges), and M is the number of light edges. □

Proof of Lemma 4.2. We need to prove three properties of $\mathcal{F}$: that $\mathcal{F}$ is a DAG, that every path in $\mathcal{F}$ reads each variable only once, and that all its nodes are labeled consistently with the number of their children (e.g., a no-op has one child). The first two properties follow from the following claim:

Claim: If u is a decision node in $\mathcal{D}$ labeled with a variable X, and there exists a non-trivial path (with at least one edge) between the nodes (u, s) (v, s') in $\mathcal{F}$, then the variable X does not occur in $\mathcal{D}_v$.

Indeed, the claim implies that $\mathcal{F}$ is acyclic, because any cycle in $\mathcal{F}$ implies a non-trivial path from some node (u, s) to itself, and obviously $X \in \mathcal{D}_u$, contradicting the claim. It also implies that every path in $\mathcal{F}$ is read-once: if a path tests a variable X twice, once at (u, s) and once at (u_1, s_1), then $X \in \mathcal{D}_{u_1}$, contradicting the claim. It remains to prove the claim.

Suppose to the contrary that there exists a node (u, s) such that u is labeled with X and there exists a path from (u, s) to (v, s') in $\mathcal{F}$ such that X occurs in $\mathcal{D}_v$. Choose v such that $\mathcal{D}_v$ is maximal; i.e., there is no path from (u, s) to some (v', s'') such that $\mathcal{D}_v \subset \mathcal{D}_{v'}$ (in the latter case replace v with v': we still have that X occurs in $\mathcal{D}_{v'}$). Consider the last edge on the path from (u, s) to (v, s') in $\mathcal{F}$:

$$(u, s), \ldots, (w, s''), (v, s') \qquad (1)$$

Observe that (w, v) is not an edge in $\mathcal{D}$ since $\mathcal{D}_v$ is maximal and since (u, v) is not an edge in $\mathcal{D}$ by the read-once property of $\mathcal{D}$; therefore, the edge from (w, s'') to (v, s') is of Type 3. Thus, there exists an AND-node z with children v_1, v, and our last edge is of the form $(w, s' \cup \{e\}), (v, s')$, where $e = (z, v_1)$ the light edge of z. We claim that $e \notin s$; i.e., it is not present at the beginning of the path in (1). If $e \in s$ then, since $s \in S(u)$, we have u, which queries X, in D_{v_1}. Together with the assumption that some node in D_v queries X, we see that descendants of the two children v_1, v of AND-node z query the same variable, contradicting the fact that $\mathcal{D}$ is a decision-DNNF. This proves $e \notin s$. On the other hand, $e \in s''$. Now consider the first node on the path in (1) where e is introduced. It can only be an edge of the form $(z, s_1), (v_1, s_1 \cup \{e\})$. But now we have a path from (u, s) to (z, s_1) with $X \in \mathcal{D}_z \supset \mathcal{D}_v$, contradicting the maximality of v. This proves the claim.

Finally, we show that all nodes in $\mathcal{F}$ are consistently labeled, *i.e.* they have the correct arity. To prove this, we only need to show that every no-op node has a single child. There are two cases: the node is (u, s) where u is an AND node in $\mathcal{D}$ (for a Type 1 edge), in which case its single child is $(v_1, s \cup \{(u, v_1)\})$; or the node is (w, s) where w is a 1-sink node and $s \neq \emptyset$ (for a Type 3 edge). In that case, let $e = (z, v)$ be the last edge in s: more precisely, if P is any path such that $s = s(P)$, then e is the last light edge on P. (This e is well defined: if $s = s(P) = s(P')$ then P and P' have the same sets of light edges, and therefore must traverse them in the same order since $\mathcal{D}$ is a DAG.) Let v' be the right child of z; then the only edge from (w, s) goes to $(v', s - \{e\})$. □

Next we prove Lemma 4.3, which completes the proof of Theorem 3.1. To prove this we will use the properties that (a) the value of the function computed by an FBDD on an input assignment is the value of the sink reached on the unique path from the root followed by the input, and (b) the value of the function computed by a decision-DNNF is the logical AND of all of the sink values reachable from the root on that assignment.

Proof of Lemma 4.3. Let Φ_D and Φ_F be the Boolean formulas computed by $\mathcal{D}$ and $\mathcal{F}$ respectively. We show that for any assignment θ to the Boolean variables, $\Phi_{\mathcal{D}}[\theta] = 0$ iff $\Phi_{\mathcal{F}}[\theta] = 0$. For the "if" direction, suppose that $\Phi_{\mathcal{F}}[\theta] = 0$. Let P be the unique root-sink path in $\mathcal{F}$ consistent with θ, which must reach a 0-sink by assumption. We will show that there exists a path P' in $\mathcal{D}$ from the root to a 0-sink that is consistent with θ. This suffices to prove that $\Phi_{\mathcal{D}}[\theta] = 0$. First, notice that if P does not contain edges of Type 3, then it automatically also translates into a path leading to a 0-sink in $\mathcal{D}$ and the claim holds. Otherwise, consider an edge of Type 3 from $(w, s \cup \{e\})$ to (v_2, s) such that (i) there exists an AND-node u with children v_1, v_2, (ii) w is a descendant of v_1, and (iii) $e = (u, v_1)$. Since the edge e must have been introduced along the path, P contains an edge of the form $(u, s'), (v_1, s' \cup \{e\})$. Remove the fragment of P between (u, s') and (v_2, s): this is also a path in $\mathcal{D}$ to a 0-sink (using the original heavy-edge (u, v_2)), with one less edge of Type 3, and the claim follows by induction.

For the "only if" part, suppose that $\Phi_{\mathcal{D}}[\theta] = 0$ and P' is a path in $\mathcal{D}$ from the root to a 0-sink node; as a warm-up, if P' has no heavy edges then it translates immediately into a path in $\mathcal{F}$ to a 0-sink. In general, we proceed as follows. Consider all paths in $\mathcal{D}$ that are consistent with θ and lead to a 0-sink node. Order them lexicographically as follows: $P'_1 < P'_2$ if, for some $k \geq 1$, P'_1 and P'_2 agree on the first $k-1$ steps, and at step k P'_1 follows the light edge (u, v_1), while P'_2 follows the heavy edge (u, v_2) of some AND-node u. Let P' be a minimal path under this order. We translate it into a path P in $\mathcal{F}$ iteratively, starting from the root r. Suppose we have translated the fragment $r \to u$ of P' into a path P in $\mathcal{F}$: $(r, \emptyset) \to (u, s)$. Consider the next edge (u, v) in P': if it is a light edge e or a neutral edge, we simply extend P with $(v, s \cup \{e\})$ or (v, s) respectively. If (u, v) is a heavy edge, let (u, v_1) be its light sibling, and let $s_1 = s \cup \{e\}$. By the minimality of P', $\Phi_{v_1}[\theta] = 1$ (otherwise we could find a consistent path to a 0-sink in $\mathcal{D}_{v_1}$). We claim that there exists a 1-sink node w in $\mathcal{D}_{v_1}$ s.t. the path P'' in $\mathcal{F}_{(v_1,s_1)}$ defined by θ leads from (v_1, s_1) to (w, s_1): the claim completes the proof of the lemma, because we simply extend P with: (u, s), (v_1, s_1), P'', (w, s_1), (v, s), where the last edge is an edge of Type 3, $(w, s \cup \{e\}), (v, s)$, completing

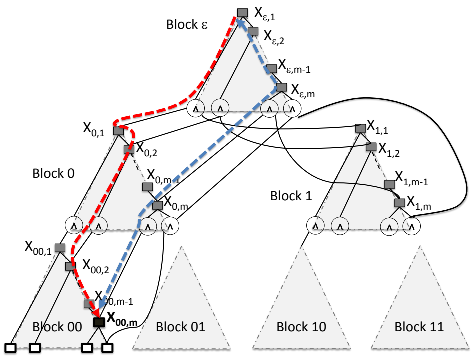


Figure 3: The decision-DNNF $D(p)$, $p = 3$, in Section 4.4. The (red and blue) bold dotted arrows denote two paths from the root to $u = X_{00,m}$. The white boxes at the lowest level denote decision nodes to 0- and 1-sinks.

our iterative construction of P.

To prove the claim, we apply our decision-DNNF-to-FBDD translation to $\mathcal{D}_{v_1}$, and let $\mathcal{F}_1$ denote the resulting FBDD; by construction, any edge (z', s'), (z'', s'') in $\mathcal{F}_1$ corresponds to an edge $(z', s' \cup s_1)$, $(z'', s'' \cup s_1)$ in $\mathcal{F}$. If $\Phi_{\mathcal{F}_1}$ is the function computed by $\mathcal{F}_1$, then we have already shown that for any θ, $\Phi_{\mathcal{F}_1}[\theta] = 0 \Rightarrow \Phi_{\mathcal{D}_{v_1}}[\theta] = 0$: for our particular θ we have $\Phi_{\mathcal{D}_{v_1}}[\theta] = \Phi_{v_1}[\theta] = 1$, hence $\Phi_{\mathcal{F}_1}[\theta] = 1$. Therefore, the path defined by θ in $\mathcal{F}_1$ goes from the root $(v_1, \emptyset)$ to some node $(w, \emptyset)$, where w is a 1-sink node in $\mathcal{D}$; the corresponding path in $\mathcal{F}_{(v_1,s_1)}$ goes from (v_1, s_1) to (w, s_1), proving the claim. □

4.4 A Tight Example

We conclude this section by showing that our analysis cannot be tightened to a polynomial bound[5] Fix $M > 0$, and let $m = M^{1/2}$. For each number $p > 0$, the decision-DNNF $\mathcal{D}_p$ given in Figure 3 (for $p = 3$) consists of $m = 2^p - 1$ blocks of size m, organized into p levels (0 to $p-1$). Each block has 2 children to the next level.

A block is identified by $w \in \{0, 1\}^*$, where $|w| \leq p-1$. Thus, $w = 011$ means "left-right-right" and $w = \epsilon$ means the root block. Each block w has $m+1$ AND-nodes, m Boolean variables ($X_{w,i}$, where $1 \leq i \leq m$), and m entry points at these m variables. The left (resp. right) child of the i-th AND-node in block w points to $X_{w0,i}$ (resp. $X_{w1,i}$), where $1 \leq i \leq m$; The left and the right children of the $(m+1)$-st AND node in block w points to the $(m+1)$-st AND node of blocks $w0$ and $w1$ respectively. Clearly, the total number of AND-nodes in the decision-DNNF is $M = m(m+1)$.

To obtain a lower bound on the size of the FBDD given by our conversion algorithm, we count the total num-

[5]This only applies to our construction. It does not separate FBDDs from decision-DNNFs, since a smaller equivalent FBDD may exist for this decision-DNNF.

ber of copies (u, s) created for the node $u = X_{00..0,m}$ (*i.e.*, the last decision node in the left-most block at the lowest level), where $s \in S(u)$ is the set of light edges on a path from the root to u. For any path P from the root to u, let $a_j < m$ be the number of consecutive decision nodes followed by P at the j-th level, for $0 \leq j \leq p-1$; P must take the left (light) branch of the corresponding AND-node at each level $j < m-1$. Note that $\sum_{j=0}^{p-1} a_j = m-1$, any choice of a_j's satisfying this corresponds to a valid path P to u, and distinct choices correspond to different sets of light edges. Therefore, $|S(u)|$ is the number of different choices of a_j which is $\binom{m-1+p}{p} \geq \binom{m}{p} \geq (m/p)^p = 2^{p(\log m - \log p)} = 2^{\Omega(\log^2 m)} = 2^{\Omega(\log^2 M)}$ since $p = \Theta(\log m)$ and $M = m(m+1)$.

5 Monotone k-DNFs

We prove Lemma 3.3 in this section. Fix a decision-DNNF $\mathcal{D}$ computing a monotone k-DNF Boolean function Φ; w.l.o.g. we assume that $\mathcal{D}$ is *non-redundant*: each child of each AND-node in $\mathcal{D}$ computes a non-constant function.

Proposition 5.1. *$\forall$ node $u \in \mathcal{D}$, Φ_u is monotone.*

Proof. The statement is true for the root node u. Suppose that Φ_u is monotone at some node u. If u is a decision node testing the variable X and with children v_0, v_1, then both $\Phi_{v_0} = \Phi_u[X = 0]$ and $\Phi_{v_1} = \Phi_u[X = 1]$ are monotone. If u is an AND-node with children u_1, u_2 then $\Phi_u = \Phi_{u_1} \wedge \Phi_{u_2}$ where Φ_{u_1}, Φ_{u_2} have disjoint sets of variables, hence they are themselves monotone. The proposition follows by induction. $\square$

In the case of a monotone function Φ, a *prime implicant* is a minimal set of variables whose conjunction implies Φ and a minimal DNF for Φ has one term for each of its prime implicants; hence, Φ can be written as k-DNF iff k is the size of its largest prime implicant. If θ is a partial assignment, then $\Phi[\theta]$ is a k'-DNF for some $k' \leq k$. Let A_u be the largest number of AND-nodes on any path from the node u to some leaf. The following proposition proves Lemma 3.3:

Lemma 5.2. *For every node u with $A_u \geq 1$, if Φ_u is a monotone k-DNF, then $k \geq A_u + 1$.*

Proof. The following claim, which we prove by induction on $|A_u|$, suffices to show the lemma: for every node u with $A_u \geq 1$, there exists a partial assignment θ such that $\Phi_u[\theta]$ is a Boolean formula that is the conjunction of $\geq A_u + 1$ variables.

Observe that it suffices to prove the claim when u is an AND-node, since for any u' with $A_{u'} \geq 1$ that is not an AND-node, there is some AND-node u reachable from u' only via decision nodes (and hence with $A_u = A_{u'}$) and we can obtain the partial assignment θ' for u' by adding the partial assignment σ determined by the path from u' to u to the partial assignment θ for u.

If u is an AND-node with children v_1, v_2, then $\Phi_u = \Phi_{v_1} \wedge \Phi_{v_2}$ where Φ_{v_1}, Φ_{v_2} do not share any variables. Consider a path starting at u that has A_u AND-nodes and assume w.l.o.g. that it takes the first branch, to v_1: thus, $A_u = A_{v_1} + 1$. If $A_{v_1} = 0$ then, since $\mathcal{D}$ is non-redundant, Φ_{v_1} is non-constant, so there is partial assignment θ_1 such that $I_1 = \Phi_{v_1}[\theta_1]$ is a conjunction of size $\geq 1 = A_{v_1} + 1$. If $A_{v_1} \geq 1$, by the induction hypothesis, there exists a partial assignment θ_1 such that $I_1 = \Phi_{v_1}[\theta_1]$ is a conjunction of size $\geq A_{v_1} + 1$. Since $\mathcal{D}$ is non-redundant, Φ_{v_2} is non-constant, so there exists a partial assignment θ_2 such that $I_2 = \Phi_{v_2}[\theta_2]$ is a conjunction of size ≥ 1. Taking $\theta = \theta_1 \cup \theta_2$ and using the disjointness of the variables in Φ_{v_1} and Φ_{v_2}, we get that $\Phi_u[\theta] = I_1 \wedge I_2$ is a conjunction of size $\geq (A_{v_1} + 1) + 1 = A_u + 1$, proving the claim. $\square$

6 Lower Bounds in Probabilistic Databases

We now show an important application of our main result to probabilistic databases. While in knowledge compilation there exists a single complexity parameter, which is the size of the input formula, in databases there are two parameters: the database query, and the database instance. For example, the query may be expressed in a query language, like SQL, and is usually very small (e.g. few lines), while the database instance is very large (e.g. billions of tuples). We are interested here in *data complexity* [Vardi, 1982], where the query is fixed, and the complexity parameter is the size of the database instance. We use Theorem 3.4 to prove an exponential lower bound for the query evaluation problem for every query that is *non-hierarchical.*

We first briefly review the key concepts in probabilistic databases, and refer the reader to [Abiteboul et al., 1995, Suciu et al., 2011] for details.

A relational vocabulary consists of k relation names, $R_1, \ldots, R_k$, where each R_i has an arity $a_i > 0$. A (deterministic) database instance is $D = (A, R_1^D, \ldots, R_k^D)$, were A is a set of constants called the *domain*, and for each i, $R_i^D \subseteq A^{a_i}$. Let $n = |A|$ be the size of the domain of the database instance.

A *Boolean query* is a function Q that takes as input a database instance D and returns an output $Q(D) \in \{\texttt{false}, \texttt{true}\}$. A *Boolean conjunctive query* (CQ) is given by an expression of the form $Q = \exists x_1 \ldots \exists x_\ell (P_1 \wedge \ldots \wedge P_m)$, where each P_k is a positive relational atom of the form $R_i(x_{p_1}, \ldots, x_{p_{a_i}})$, with x_j either a variable $\in \{x_1, \ldots, x_\ell\}$ or a constant. A *Boolean Union of Conjunctive Queries* (UCQ) is given by an expression $Q = Q_1 \vee \ldots \vee Q_m$ where each Q_i is a

Patient P		
name	disease	
Ann	asthma	X_1
Bob	asthma	X_2
Carl	flue	X_3

Friend F		
name1	name2	
Ann	Joe	Z_{11}
Ann	Tom	Z_{12}
Bob	Tom	Z_{22}
Carl	Tom	Z_{32}

Smoker S	
name	
Joe	Y_1
Tom	Y_2

Query $Q = \exists x \; \exists y \; \texttt{P}(x, \texttt{'asthma'}) \; \wedge \; \texttt{F}(x, y) \; \wedge \; \texttt{S}(y)$

Lineage expression $\Phi_Q^D = X_1 Z_{11} Y_1 \vee X_1 Z_{12} Y_2 \vee X_2 Z_{22} Y_2$

Figure 4: A database instance, query, and lineage

Boolean conjunctive query. We assume all queries to be minimized (i.e. they do not have redundant atoms, see [Abiteboul et al., 1995]).

Given an instance D and query expression Q, the *lineage* Φ_Q^D is a Boolean formula obtained by grounding the atoms in Q with tuples in D; it is similar to *grounding* in knowledge representation [Domingos and Lowd, 2009]. Formally, each tuple t in the database D is associated with unique Boolean variable X_t, and the lineage is defined inductively on the structure of Q: (1) $\Phi_Q^D = X_t$, if Q is the ground tuple t, (2) $\Phi_{Q_1 \wedge Q_2}^D = \Phi_{Q_1}^D \wedge \Phi_{Q_2}^D$, (3) $\Phi_{Q_1 \vee Q_2}^D = \Phi_{Q_1}^D \vee \Phi_{Q_2}^D$, and (4) $\Phi_{\exists x.Q}^D = \bigvee_{a \in A} \Phi_{Q[a/x]}^D$. The lineage is always a monotone k-DNF of size $O(n^\ell)$, where n is the domain size and k, ℓ are the largest number of atoms, and the largest number of variables in any conjunctive query Q_i of Q.

In a *probabilistic database* [Suciu et al., 2011], every tuple t in the database instance is uncertain, and the Boolean variable X_t indicates whether t is present or not. The probability $P(X_t = \texttt{true})$ is known for every tuple t, and is stored in the database as a separate attribute of the tuple. The goal in probabilistic databases is: given a query Q and an input database D, compute the probability of its lineage, $P(\Phi_Q^D)$.

Example 6.1. *The following example is adapted from [Jha et al., 2010], on a vocabulary with three relations* `Patient(name, diseases)`, `Friend(name1, name2)`, `Smoker(name)` *(see Figure 4). Each tuple is associated with a Boolean variable (X_1, X_2 etc). The Boolean conjunctive query Q (as well as the lineage Φ_Q^D for database D) returns* `true` *if the database instance contains an asthma patient who has a smoker friend. Our goal is to compute $P(\Phi_Q^D)$, given the probabilities of each Boolean variable, when Q is fixed and D is variable.*

Lemma 6.2. *Let h be the conjunctive query $\exists x \exists y R(x) \wedge S(x, y) \wedge T(y)$. Then any decision-DNNF for the Boolean formula Φ_h^D has size $2^{\Omega(\sqrt{n})}$, where n is the size of the domain of D.*

Proof. For any n, let D be the database instance $R^D = [n]$, $S^D = [n] \times [n]$, $T^D = [n]$. Then the lineage Φ_h^D is exactly the formula Φ_n of Corollary 3.6, up to variable renaming, and the claim follows. □

Fix a conjunctive query $q = \exists x_1 \dots \exists x_\ell P_1 \wedge \dots \wedge P_m$. For each variable x_j, let $at(x_j)$ denote the set of atoms P_i that contain the variable x_j.

Definition 6.3. *[Suciu et al., 2011] The query q is called* hierarchical *if for any two distinct variables x_i, x_j, one of the following holds: $at(x_i) \subseteq at(x_j)$, or $at(x_i) \supseteq at(x_j)$, or $at(x_i) \cap at(x_j) = \emptyset$. A Boolean Union of Conjunctive Queries $Q = q_1 \vee \dots \vee q_k$ is called* hierarchical *if every q_i is hierarchical for $i \in [k]$.*

For example, the query h in Lemma 6.2 is non-hierarchical, because $at(x) = \{R, S\}$, $at(y) = \{S, T\}$, while the query $\exists x. \exists y. R(x) \wedge S(x, y)$ is hierarchical. It is known that, for a non-hierarchical UCQ Q, computing the probability of the Boolean formulas Φ_Q^D is #P-hard [Suciu et al., 2011]. In the full paper we prove:

Theorem 6.4. *Consider any Boolean Union of Conjunctive Queries Q. If Q is non-hierarchical, then the size of the decision-DNNF for the Boolean functions Φ_Q^D is $2^{\Omega(\sqrt{n})}$, where n is the size of the domain of D.*

The query Q in Example 6.1 is non-hierarchical: $at(x) = \{\texttt{Patient}, \texttt{Friend}\}$ and $at(y) = \{\texttt{Friend}, \texttt{Smoker}\}$. Therefore, any decision-DNNF computing Q has size $2^{\Omega(\sqrt{n})}$. For another example, consider the following non-hierarchical query that returns true iff the database contains a triangle of friends:

$$q' = \exists x \exists y \exists z (\texttt{F}(x, y) \wedge \texttt{F}(y, z) \wedge \texttt{F}(z, x))$$

Its lineage is $\Delta_n = \bigvee_{i,j,k=1,n} Z_{ij} Z_{jk} Z_{ki}$. By Theorem 6.4, any decision-DNNF for Δ_n has size $2^{\Omega(\sqrt{n})}$.

7 Conclusions and Open Problems

We have proved that any decision-DNNF can be efficiently converted into an equivalent FBDD that is at most quasipolynomially larger, and at most polynomially larger in the case of k-DNF formulas. As a consequence, known lower bounds for FBDDs imply lower bounds for decision-DNNFs and thus (a) exponential separations of the representational power of decision-DNNFs from that of both d-DNNFs and AND-FBDDs and (b) lower bounds on the running time of any algorithm that, either explicitly or implicitly, produces a decision-DNNF, including the current generation of exact model counting algorithms.

Some natural questions arise: Is there a polynomial simulation of decision-DNNFs by FBDDs for the general case? In particular, is there a polynomial-size FBDD for the example Section 4.4? Is there some other, more powerful syntactic subclass of d-DNNFs that is useful for exact model counting? What can be said about the limits of *approximate* model counting [Gomes et al., 2009]?

References

[Abiteboul et al., 1995] Abiteboul, S., Hull, R., and Vianu, V. (1995). *Foundations of Databases.*

[Akers, 1978] Akers, S. B. (1978). Binary decision diagrams. *IEEE Trans. Computers*, 27(6):509–516.

[Bacchus et al., 2003] Bacchus, F., Dalmao, S., and Pitassi, T. (2003). Algorithms and complexity results for #sat and bayesian inference. In *FOCS*, pages 340–351.

[Bayardo et al., 2000] Bayardo, R. J., Jr., and Pehoushek, J. D. (2000). Counting models using connected components. In *AAAI*, pages 157–162.

[Beame et al., 2010] Beame, P., Impagliazzo, R., Pitassi, T., and Segerlind, N. (2010). Formula caching in dpll. *TOCT*, 1(3).

[Beame et al., 2004] Beame, P., Kautz, H. A., and Sabharwal, A. (2004). Towards understanding and harnessing the potential of clause learning. *J. Artif. Intell. Res. (JAIR)*, 22:319–351.

[Ben-Sasson and Wigderson, 2001] Ben-Sasson, E. and Wigderson, A. (2001). Short proofs are narrow – resolution made simple. *J. ACM*, 48(2):149–169.

[Bollig and Wegener, 1998] Bollig, B. and Wegener, I. (1998). A very simple function that requires exponential size read-once branching programs. *Inf. Process. Lett.*, 66(2):53–57.

[Bryant, 1986] Bryant, R. E. (1986). Graph-based algorithms for boolean function manipulation. *IEEE Trans. Computers*, 35(8):677–691.

[Darwiche, 2001a] Darwiche, A. (2001a). Decomposable negation normal form. *J. ACM*, 48(4):608–647.

[Darwiche, 2001b] Darwiche, A. (2001b). On the tractable counting of theory models and its application to truth maintenance and belief revision. *Journal of Applied Non-Classical Logics*, 11(1-2):11–34.

[Darwiche and Marquis, 2002] Darwiche, A. and Marquis, P. (2002). A knowledge compilation map. *J. Artif. Int. Res.*, 17(1):229–264.

[Davis et al., 1962] Davis, M., Logemann, G., and Loveland, D. (1962). A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397.

[Davis and Putnam, 1960] Davis, M. and Putnam, H. (1960). A computing procedure for quantification theory. *J. ACM*, 7(3):201–215.

[Domingos and Lowd, 2009] Domingos, P. and Lowd, D. (2009). *Markov Logic: An Interface Layer for Artificial Intelligence.*

[Gomes et al., 2009] Gomes, C. P., Sabharwal, A., and Selman, B. (2009). Model counting. In *Handbook of Satisfiability*, pages 633–654.

[Huang and Darwiche, 2005] Huang, J. and Darwiche, A. (2005). Dpll with a trace: From sat to knowledge compilation. In *IJCAI*, pages 156–162.

[Huang and Darwiche, 2007] Huang, J. and Darwiche, A. (2007). The language of search. *JAIR*, 29:191–219.

[Jha et al., 2010] Jha, A. K., Gogate, V., Meliou, A., and Suciu, D. (2010). Lifted inference seen from the other side : The tractable features. In *NIPS*, pages 973–981.

[Majercik and Littman, 1998] Majercik, S. M. and Littman, M. L. (1998). Using caching to solve larger probabilistic planning problems. In *AAAI*, pages 954–959.

[Masek, 1976] Masek, W. J. (1976). *A fast algorithm for the string editing problem and decision graph complexity.* Master's thesis, MIT.

[Muise et al., 2012] Muise, C., McIlraith, S. A., Beck, J. C., and Hsu, E. I. (2012). Dsharp: fast d-dnnf compilation with sharpsat. In *Canadian AI*, pages 356–361.

[Sang et al., 2004] Sang, T., Bacchus, F., Beame, P., Kautz, H. A., and Pitassi, T. (2004). Combining component caching and clause learning for effective model counting. In *SAT.*

[Suciu et al., 2011] Suciu, D., Olteanu, D., Ré, C., and Koch, C. (2011). *Probabilistic Databases.*

[Thurley, 2006] Thurley, M. (2006). sharpsat: counting models with advanced component caching and implicit bcp. In *SAT*, pages 424–429.

[Vardi, 1982] Vardi, M. Y. (1982). The complexity of relational query languages. In *STOC*, pages 137–146.

[Wegener, 2000] Wegener, I. (2000). *Branching programs and binary decision diagrams: theory and applications.*

Reasoning about Probabilities in Dynamic Systems using Goal Regression[*]

Vaishak Belle and **Hector J. Levesque**
Dept. of Computer Science
University of Toronto
Toronto, Ontario M5S 3H5, Canada
{vaishak, hector}@cs.toronto.edu

Abstract

Reasoning about degrees of belief in uncertain dynamic worlds is fundamental to many applications, such as robotics and planning, where actions modify state properties and sensors provide measurements, both of which are prone to noise. With the exception of limited cases such as Gaussian processes over linear phenomena, belief state evolution can be complex and hard to reason with in a general way. This paper proposes a framework with new results that allows the reduction of subjective probabilities after sensing and acting, both in discrete and continuous domains, to questions about the initial state only. We build on an expressive probabilistic first-order logical account by Bacchus, Halpern and Levesque, resulting in a methodology that, in principle, can be coupled with a variety of existing inference solutions.

1 INTRODUCTION

Reasoning about *degrees of belief* in uncertain dynamic worlds is fundamental to many applications, such as robotics and planning, where actions modify state properties and sensors provide measurements, both of which are prone to noise. However, there seem to be two disparate paradigms to address this concern, both of which have their limitations. At one extreme, there are logical formalisms, such as the *situation calculus* (McCarthy and Hayes, 1969; Reiter, 2001), which allows us to express strict uncertainty, and exploits regularities in the effects actions have on propositions to describe physical laws compactly. Probabilistic sensor fusion, however, has received less attention here. At the other extreme, revising beliefs after noisy observations over rich error profiles is effortlessly addressed using *probabilistic techniques* such as Kalman filtering and Dynamic Bayesian Networks (Dean and Kanazawa, 1989; Dean and Wellman, 1991). However, in these frameworks, a complete specification of the dependencies between variables is taken as given, making it difficult to deal with other forms of incomplete knowledge as well as complex actions that shift dependencies between variables in nontrivial ways.

An influential but nevertheless simple proposal by Bacchus, Halpern and Levesque (1999), BHL henceforth, was among the first to merge these broad areas in a general way. Their specification is widely applicable because it is not constrained to particular structural assumptions. In a nutshell, they extend the situation calculus language with a provision for specifying the degrees of belief in formulas in the initial state, closely fashioned after intuitions on incorporating probability in modal logics (Halpern, 1990; Fagin and Halpern, 1994). This then allows incomplete and partial specifications, which might be compatible with one or very many initial distributions and sets of independence assumptions, with beliefs following at a corresponding level of specificity. Moreover, together with a rich action theory, the model not only exhibits Bayesian conditioning (Pearl, 1988) (which, then, captures special cases such as Kalman filtering), but also allows flexibility in the ways dependencies and distributions may change over actions.

What is left open, however, is the following computational concern: how do we effectively reason about degrees of belief in the framework? That is, while changing degrees of belief do indeed emerge as *logical entailments* of the given action theory, no procedure is given for computing these entailments. On closer examination, in fact, this is a two-part question:

(i) How do we effectively reason about beliefs in a particular state?
(ii) How do we effectively reason about belief state evolution and belief change?

In the simplest case, part (i) puts aside acting and sensing, and considers reasoning about the *initial* state only, which is then the classical problem of (first-order) probabilistic inference. We do not attempt to do a full survey here, but this has received a lot of attention, often under reasonable assumptions such as the ability to factorize domains (Poole, 2003; Gogate and Domingos, 2010).

[*]We thank the reviewers for very helpful comments. Financial support from the Natural Science and Engineering Research Council of Canada made this research possible.

This paper is about part (ii). Addressing this concern has a critical bearing on the assumptions made about the domain for tractability purposes. For example, if the initial state supports a decomposed representation of the distribution, can we expect the same after actions? In the exception of very limited cases such as Kalman filtering that harness the conjugate property of Gaussian processes, the situation is discouraging. In fact, even in the slightly more general case of Dynamic Bayesian Networks, which are in essence atomic propositions, if one were to assume that state variables are independent at time 0, they can become fully correlated after a few steps (Dean and Kanazawa, 1988; Boyen and Koller, 1998; Hajishirzi and Amir, 2010). Dealing with complex actions, incomplete specifications and mixed representations, therefore, is significantly more involved.

In this paper, we propose a new alternative to infer degrees of belief in the presence of a rich theory of actions, closely related to *goal regression* (Waldinger, 1977; Reiter, 2001). The procedure is general, not requiring (but allowing) structural constraints about the domain, nor imposing (but allowing) limitations to the family of actions. Regression derives a mathematical formula, using term and formula *substitution* only, that relates belief after a sequence of actions and observations, even when they are noisy, to beliefs about the initial state. That is, among other things, if the initial state supports efficient factorizations, regression will maintain this advantage no matter how actions affect the dependencies between state variables over time. Going further, the formalism will work seamlessly with discrete probability distributions, probability densities, and perhaps most significantly, with difficult combinations of the two. (See Example 9.3 in Section 4 below.)

To see a simple example of what goal regression does, imagine a robot facing a wall and at a certain distance h to it, as in Figure 1. The robot might initially believe h to be drawn from a uniform distribution on $[2, 12]$. Assume the robot moves away by 2 units and is now interested in the belief about $h \leq 5$. Regression would tell the robot that this is equivalent to its initial beliefs about $h \leq 3$ which here would lead to a value of .1. To see a nontrivial example, imagine now the robot is also equipped with a sonar unit aimed at the wall, that adds Gaussian noise with mean μ and variance σ^2. After moving away by 2 units, if the sonar were now to provide a reading of 8, then regression would derive that belief about $h \leq 5$ is equivalent to

$$\frac{1}{\gamma} \int_2^3 .1 \times \mathcal{N}(6 - x; \mu, \sigma^2)\, \mathrm{d}x.$$

where γ is the normalization factor. Essentially, the posterior belief about $h \leq 5$ is reformulated as the product of the prior belief about $h \leq 3$ and the likelihood of $h \leq 3$ given an observation of 6. (That is, observing 8 after moving away by 2 units is related here to observing 6 initially.)

We believe the broader contributions of this line of work are

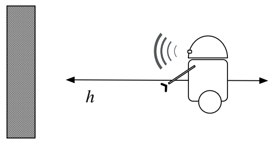


Figure 1: Robot moving towards a wall.

two-fold. On the one hand, as we show later, simple cases of belief state evolution, as applicable to conjugate distributions for example, are special cases of regression's backward chaining procedure. Thus, regression could serve as a *formal basis* to study probabilistic belief change wrt limited forms of actions. On the other hand, our contribution can be viewed as a methodology for combining actions with recent advances in probabilistic inference, because reasoning about actions reduces to reasoning about the initial state.

We now describe the structure of the paper. Before moving on, we note that although the original BHL account is only suitable for discrete domains (as they assume values are taken from countable sets), in a companion paper (Belle and Levesque, 2013a) we show that the account can be generalized to domains with both discrete and continuous variables with minimal additions. In the preliminaries section, we cover the situation calculus, recap BHL and go over the essentials of its continuous extension. We then present regression for discrete domains, followed by regression for general domains. We end with related and future work. For this version of the paper, we allow noisy sensors but assume deterministic (noise-free) physical actions. Noisy actions are left for an extended version.

2 BACKGROUND

The language $\mathcal{L}$ of the situation calculus (Reiter, 2001) is a many-sorted dialect of predicate calculus, with sorts for *physical actions*, *sensing actions, situations* and *objects* (including the set of reals $\mathbb{R}$ as a subsort). A situation represents a history as a sequence of actions. A set of initial situations correspond to the ways the world might be initially. Successor situations are the result of doing actions, where the term $do(a, s)$ denotes the unique situation obtained on doing a in s. The term $do(\alpha, s)$, where α is the sequence $[a_1, \ldots, a_n]$, abbreviates $do(a_n, do(\ldots, do(a_1, s) \ldots))$. For example, $do([grasp(o_1), repair(o_1)], s)$ represents the situation obtained after grasping and repairing object o_1 starting from s. Initial situations are those without a predecessor:

$$Init(s) \doteq \neg\exists a, s'.\ s = do(a, s').$$

We let the constant S_0 denote the actual initial situation, and we use the variable ι to range over initial situations only. $\mathcal{L}$ also includes functions whose values vary from situation to situation, called *fluents*, whose last argument is a situation.

We follow two notational conventions. We often suppress the situation argument in a formula ϕ, or use a distinguished variable *now*. Either way, $\phi[t]$ is used to denote the formula with that variable replaced by t, *e.g.* both $(f < 12)[s]$ and $(f(now) < 12)[s]$ mean $f(s) < 12$. We also use conditional *if-then-else* expressions in formulas throughout. We write $f = \text{If } \phi \text{ Then } t_1 \text{ Else } t_2$ to mean $[\phi \wedge f = t_1] \vee [\neg\phi \wedge f = t_2]$. In case quantifiers appear inside the *if*-condition, we take some liberties with notation and the scope of variables in that we write $f = \text{If } \exists x.\ \phi \text{ Then } t_1 \text{ Else } t_2$ to mean $\exists x\ [\phi \wedge f = t_1] \vee [(f = t_2) \wedge \neg\exists x.\ \phi]$.

Basic action theory

Following (Reiter, 2001), we model dynamic domains in $\mathcal{L}$ by means of a *basic action theory* $\mathcal{D}$, which consists of domain-independent *foundational* axioms, *unique name* axioms for actions (see (Reiter, 2001)), and (1) axioms $\mathcal{D}_0$ that describe what is true in the initial states, including S_0;[1] (2) precondition axioms[2] of the form $Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s)$ describing executability conditions using a special fluent *Poss*; and (3) successor state axioms of the following form stipulating how fluents change:

$$\begin{aligned} f(do(a, s)) = u \equiv\ & \exists\vec{z_1}[a = A_1(\vec{z_1}) \wedge u = e_1(\vec{z_1}, s)] \vee \ldots \\ & \exists\vec{z_k}[a = A_k(\vec{z_k}) \wedge u = e_k(\vec{z_k}, s)] \vee \\ & \bigwedge \neg\exists\vec{z_i}[a = A_i(\vec{z_i})] \wedge u = f(s). \end{aligned} \tag{1}$$

where $e_i(\vec{z_i}, s)$ is any expression whose only free variables are $\vec{z_i}$ and s. For example, consider the action $fwd(z)$ of moving precisely z units towards or away from the wall, but the motion stops when the wall is reached:

$$\begin{aligned} h(do(a, s)) = u \equiv\ & \exists z[a = fwd(z) \wedge u = \max(0, h(s) - z)] \vee \\ & \neg\exists z[a = fwd(z)] \wedge u = h(s). \end{aligned} \tag{2}$$

This sentence states that $fwd(z)$ is the only action affecting fluent h, in effect incorporating a solution to the frame problem (Reiter, 2001). Given an action theory, an agent reasons about actions by means of entailments of $\mathcal{D}$.[3]

Likelihood and belief

The BHL model of belief builds on a treatment of *knowledge* in $\mathcal{L}$ (Scherl and Levesque, 2003). Here we present a simpler variant based on two distinguished fluents l and p.

The term $l(a, s)$ is intended to denote the likelihood of action a in situation s. For example, suppose $sonar(z)$ is the action of reading the value z from a sonar that measures the distance to the wall, h. We might assume that this action is characterized by a simple discrete error model (continuous error models are considered later):

$$l(sonar(z), s) = \text{If } |h(s) - z| \leq 1 \text{ Then } 1/3 \text{ Else } 0 \tag{3}$$

which stipulates that the difference between a reading of z and the true value h is either $\{0, -1, 1\}$ with probability $1/3$, assuming that h and z take integer values. In general, the action theory $\mathcal{D}$ is assumed to contain for each sensor $sense_i(\vec{x})$ that measures a fluent f, an axiom of the form:

$$l(sense_i(\vec{x}), s) = Err_i(\vec{x}, f(s)),$$

where $Err_i(u_1, u_2)$ is some expression with only two free variables u_1 and u_2, both numeric.[4] (Noise-free physical actions are given a likelihood of 1.)

Next, the p fluent determines a (subjective) probability distribution on situations. The term $p(s', s)$ denotes the relative *weight* accorded to situation s' when the agent happens to be in situation s, as in modal probability logics (Fagin and Halpern, 1994). Now, the task of the modeler is to specify the initial properties of p as part of $\mathcal{D}_0$ using ι and S_0, *e.g.*:

$$p(\iota, S_0) = \text{If } h(\iota) \in \{2, \ldots, 11\} \text{ Then } .1 \text{ Else } 0 \tag{4}$$

says that h is drawn from a uniform distribution. The following nonnegative constraint is also included in $\mathcal{D}_0$:

$$\forall\iota, s.\ p(s, \iota) \geq 0 \wedge (p(s, \iota) > 0 \supset Init(s)) \tag{P1}$$

Then, by means of a remarkably simple successor state axiom for p, (P2) below, the formal specification is complete.

$$\begin{aligned} p(s', do(a, s)) = & \\ & \text{If } \exists s''.\ s' = do(a, s'') \wedge Poss(a, s'') \\ & \quad \text{Then } p(s'', s) \times l(a, s'') \\ & \text{Else } 0 \end{aligned} \tag{P2}$$

In particular, the *degree of belief* in a formula ϕ can be accounted for in terms of an abbreviation:[5]

$$Bel(\phi, s) \doteq \frac{1}{\gamma} \sum_{\{s' : \phi[s']\}} p(s', s) \tag{B}$$

where γ, the normalization factor, is understood throughout as the same expression as the numerator but with ϕ replaced by *true*, *e.g.* here γ is $\sum_{s'} p(s', s)$. So, as in probability logics, belief is simply the total weight of worlds satisfying ϕ. But the novelty here is that in a dynamical setting, belief change via (B) is identical to Bayesian conditioning:

[1]Note that $\mathcal{D}_0$ can include any (classical) first-order sentence about S_0, such as $h(S_0) > 12$ and $f_1(S_0) \neq 2 \vee f_2(S_0) = 5$.

[2]Free variables in any of these axioms should be understood as universally quantified from the outside.

[3]Entailments are wrt standard Tarskian models, but we will also assume that models assign the usual interpretations to $=, <, >, 0, 1, +, \times, /, -, e, \pi$, and x^y (exponentials).

[4]This captures the idea that the error model of a sensor measuring f depends only on the true value of f, and is independent of other factors. In a sense this follows the Bayesian model that conditioning on a random variable f is the same as conditioning on the event of observing f. But this is not required in general in the BHL scheme, an issue we ignore for this paper.

[5]Summations can be expressed as logical terms. See BHL.

Proposition 1: *Suppose $\mathcal{D}$ includes* (P1), (P2) *and the likelihood axiom for a sensor sense(z) measuring f. Then*

$$\mathcal{D} \models Bel(f = t, do(sense(z), S_0)) = \frac{Bel(f = t, S_0) \cdot Err(z, t)}{\sum_x Bel(f = x, S_0) \cdot Err(z, x)}$$

Essentially, if the robot's sensors are informative, in the sense of returning values closer to the true value, beliefs are strengthened over time.

From sums to integrals

While the definition of belief in BHL has many desirable properties, it is defined in terms of a *summation* over situations, and therefore precludes fluents whose values range over the reals. The continuous analogue of (B) then requires *integrating* over some suitable space of values.

As it turns out, a suitable space can be found. First, assume that there are n fluents $f_1, \ldots, f_n$ in $\mathcal{L}$, and that these take no arguments other than the situation argument.[6] Next, suppose that that there is exactly one initial situation for every possible value of these fluents (Levesque *et al.*, 1998):

$$[\forall \vec{x} \exists \iota \bigwedge f_i(\iota) = x_i] \wedge [\forall \iota, \iota'. \bigwedge f_i(\iota) = f_i(\iota') \supset \iota = \iota'] \quad \text{(I)}$$

Under these assumptions, it can be shown that the summation over all situations in (B) can be recast as a summation over all possible initial values $x_1, \ldots, x_n$ for the fluents:

$$Bel(\phi, s) \doteq \frac{1}{\gamma} \sum_{\vec{x}} P(\vec{x}, \phi, s) \qquad \text{(B}')$$

where $P(\vec{x}, \phi, s)$ is the (unnormalized) weight accorded to the *successor* of an initial world where f_i equals x_i:

$$\begin{aligned} &P(\vec{x}, \phi, do(\alpha, S_0)) \doteq \\ &\quad \textsc{If} \ \exists \iota. \bigwedge f_i(\iota) = x_i \wedge \phi[do(\alpha, \iota)] \\ &\quad\quad \textsc{Then} \ p(do(\alpha, \iota), do(\alpha, S_0)) \\ &\quad \textsc{Else} \ 0 \end{aligned}$$

for any action sequence α. In a nutshell, because every situation has an initial situation as an ancestor, and because there is a bijection between initial situations and possible fluent values, it is sufficient to sum over fluent values to obtain the belief even for non-initial situations. Note that unlike (B), this one expects the final situation term $do(\alpha, S_0)$ mentioning what actions and observations took place to be explicitly specified, but that is just what one expects when the agent reasons about its belief after acting and sensing.

The generalization to the continuous case then proceeds as follows. First, we observe that some (though possibly not all) fluents will be real-valued, and that $p(s', s)$ will now be a measure of *density* not weight. For example, if h is real-valued, we might have the following analogue to (4):

$$p(\iota, S_0) = \textsc{If} \ 2 \leq h(\iota) \leq 12 \ \textsc{Then} \ .1 \ \textsc{Else} \ 0 \qquad (5)$$

which says that the true initial value of h is drawn from a uniform distribution on [2,12]. Similarly, the P term above now measures (unnormalized) density rather than weight.

Now suppose fluents are partitioned into two groups: the first k take their values $x_1, \ldots, x_k$ from $\mathbb{R}$, while the rest take their values $y_{k+1}, \ldots, y_n$ from countable domains, then the *degree of belief* in ϕ is an abbreviation for:

$$Bel(\phi, s) \doteq \frac{1}{\gamma} \int_{\vec{x}} \sum_{\vec{y}} P(\vec{x} \cdot \vec{y}, \phi, s) \qquad \text{(B}^+)$$

That is, the belief in ϕ is obtained by ranging over all possible fluent values, and integrating[7] and summing the densities of *successor* situations where ϕ holds.[8]

To summarize the formalization, a basic action theory $\mathcal{D}$ henceforth is assumed to additionally include: (a) (P1) and (I) as part of $\mathcal{D}_0$; (b) (P2) as part of $\mathcal{D}$'s successor state axioms, and (c) sensor likelihood axioms.

3 REGRESSION FOR DISCRETE DOMAINS

We now investigate a computational mechanism for reasoning about beliefs after a trajectory. In this section, we focus on discrete domains, where a *weight*-based notion of belief would be appropriate. Domains with both discrete and continuous variables are reserved for the next section.

Formally, given a basic action theory $\mathcal{D}$, a sequence of actions α, we might want to determine whether a formula ϕ *holds* after executing α starting from S_0:

$$\mathcal{D} \models \phi[do(\alpha, S_0)] \qquad (6)$$

which is called *projection* (Reiter, 2001). When it comes to beliefs, and in particular how that changes after acting and sensing, we might be interested in *calculating* the degrees of belief in ϕ after α: find a real number n such that

$$\mathcal{D} \models Bel(\phi, do(\alpha, S_0)) = n. \qquad (7)$$

The obvious method for answering (6) is to translate both $\mathcal{D}$ and ϕ into a predicate logic formula. This approach, however, presents a serious computational problem because belief formulas expand into a large number of sentences using (P2), resulting in an enormous search space with initial and successor situations. The other issue with

[6]It might be desirable to have fluents take arguments other than the situation. See (Belle and Levesque, 2013a) for discussions.

[7]Like in BHL, where summations are captured as logical terms using second-order quantification, we can use logical formulas to capture a variety of sorts of integrals. See (Belle and Levesque, 2013a). We will henceforth simply suppose that for any term t and variable x, $\int_x t$ is a term which evaluates (in the standard calculus sense) to the integral of t between $[-\infty, \infty]$.

[8]We are assuming here that the density function is (Riemann) integrable. If it is not or if $\gamma = 0$ then belief is clearly not defined, nor should it be.

this approach is that sums (and integrals in the continuous case) reduce to complicated second-order formulas.

We now introduce a *regression* procedure to simplify both (6) and (7) to queries about $Bel(\phi, S_0)$, over arithmetic expressions, for which standard probabilistic reasoning methods can be applied. For this purpose, in the sequel, *Bel* is treated as a *special syntactic operator* rather than as an abbreviation for other formulas. To see a simple example of the procedure, imagine the robot is interested in the probability of $h=7$, given (4), after reading 5 from a sonar:

$$Bel(h = 7, do(sonar(5), S_0)) \tag{8}$$

If we are to take the sonar's model to be (3), then (8) should be 0 by Bayesian conditioning because the likelihood of the true value being 7 given an observation of 5 is 0. Regression would reduce the term (8) to one over initial priors:

$$\frac{1}{\gamma} \sum_{x \in \{2,\dots,11\}} Err(5, x) \times Bel(h = x \wedge h = 7, S_0) \tag{9}$$

where *Err* is the error model from (3). By the condition inside *Bel*, the only valid value for x is 7 for which the prior is .1 but $Err(5, 7)$ is 0. Thus, (8) = (9) = 0. In general, regression is a *recursive* procedure that works iteratively over a sequence of actions discarding one action at a time, and it can be utilized to measure any logical property about the variables, *e.g.* $2\pi \cdot h < 12, h/fuel \le mileage$, *etc.*

Formally, regression operates at two levels. (Note that this differs slightly from (Reiter, 2001; Scherl and Levesque, 2003).) At the formula level,[9] we introduce an operator $\mathcal{R}$ for regressing formulas, which over equality literals sends the individual terms to an operator $\mathcal{T}$ for regressing terms. The fundamental objective of these operators is eliminate *do* symbols. The end result, then, is to transform any expression whose situation term is a successor of S_0, say $do([a_1, a_2], S_0)$, to one about S_0 only, at which point $\mathcal{D}_0$ is all that is needed. As hinted earlier, these operators treat $Bel(\phi, s)$ as though they are special sorts of terms.[10] Throughout the presentation, we assume that the inputs to these operators do not quantify over all situations.

Definition 2: For any term t, we inductively define $\mathcal{T}[t]$:

1. If t is situation-independent (*e.g.* $x, \pi^{2/3}$) then $\mathcal{T}[t] = t$.

2. $\mathcal{T}[g(t_1, \dots, t_k)] = g(\mathcal{T}[t_1], \dots, \mathcal{T}[t_k])$,
   where g is any non-fluent function (*e.g.* $\times, +, \mathcal{N}$).

3. For a fluent function f, $\mathcal{T}[f(s)]$ is defined inductively
   (a) if s is of the form $do(A(\vec{t}), s')$ then $\mathcal{T}[f(s)] = \mathcal{T}[e(\vec{t}, s')]$
   (b) else $\mathcal{T}[f(s)] = f(s)$

   where, in (a), an appropriate instance of the *rhs* of the successor state axiom is used, as obtained from (1).

4. $\mathcal{T}[Bel(\phi, s)]$ is defined inductively:
   (a) if s is of the form $do(a, s')$ and a is a physical action, then
   $$\mathcal{T}[Bel(\phi, s)] = \mathcal{T}[Bel(\psi, s')]$$
   where ψ is $Poss(a, now) \supset \mathcal{R}[\phi[do(a, now)]]$.
   (b) if s is of the form $do(a, s')$ and a is a sensing action $sense(z)$ such that $l(sense(z), s) = Err(z, f_i(s))$ is in $\mathcal{D}$ then
   $$\mathcal{T}[Bel(\phi, s)] = \frac{1}{\gamma} \sum_{x_i} Err(z, x_i) \times \mathcal{T}[Bel(\psi, s')]$$
   where ψ is $Poss(a, now) \supset \phi \wedge f_i(now) = x_i$, and γ is the normalization factor and is the same expression as the numerator but ϕ replaced by *true*.
   (c) else $\mathcal{T}[Bel(\phi, s)] = Bel(\phi, s)$.

Definition 3: For any formula ϕ, we define $\mathcal{R}[\phi]$ inductively:

1. $\mathcal{R}[t_1 = t_2] = (\mathcal{T}[t_1] = \mathcal{T}[t_2])$

2. $\mathcal{R}[G(t_1, \dots, t_k)] = G(\mathcal{T}[t_1], \dots, \mathcal{T}[t_k])$
   where G is any non-fluent predicate (*e.g.* $=, <$).

3. $\mathcal{R}[Poss(A(\vec{t}), s)] = \mathcal{R}[\Pi_A(\vec{t}, s)]$,
   where an appropriate instance of the *rhs* of the precondition axiom replaces the atom (see Section 2).

4. When ψ is a formula, $\mathcal{R}[\neg\psi] = \neg\mathcal{R}[\psi]$,
   $\mathcal{R}[\forall x \psi] = \forall x \mathcal{R}[\psi]$, $\mathcal{R}[\exists x \psi] = \exists x \mathcal{R}[\psi]$.

5. When ψ_1 and ψ_2 are formulas,
   $\mathcal{R}[\psi_1 \wedge \psi_2] = \mathcal{R}[\psi_1] \wedge \mathcal{R}[\psi_2]$,
   $\mathcal{R}[\psi_1 \vee \psi_2] = \mathcal{R}[\psi_1] \vee \mathcal{R}[\psi_2]$.

This completes the definition of $\mathcal{T}$ and $\mathcal{R}$. We now go over the justifications for the items, starting with the operator $\mathcal{T}$. In item 1, non-fluents simply do not change after actions. In item 2, $\mathcal{T}$ operates over sums and products in a modular manner. In item 3, provided there are remaining *do* symbols, the physics of the domain determines what the conditions must have been in the previous situation for the current value to hold. In item 4, if there is a remainder physical action, part (a) says that belief in ϕ after actions is simply the prior belief about the regression of ϕ, contingent on action executability. Part (b) says that the belief about

[9] For simplicity, in what follows, functional fluents in formulas are only allowed to occur as arguments of an equality literal. It is easy to show that every sentence can be transformed into an equivalent one in the required form, and the transformation is linear in the size of the original sentence, *e.g.* $h \le 9$ is written as $\exists u\,(h = u \wedge u \le 9)$.

[10] In the context of *Bel*, ϕ is understood to be any situation-suppressed formula not mentioning p, l and *Bel*. If situation terms do appear in ϕ, then they may only be the distinguished variable *now*.

ϕ after observing z for the true value of f_i is the prior belief for all possible values x_i for f_i that agree with ϕ, times the likelihood of f_i being x_i given z. The appropriateness of parts (a) and (b) depend on the fact that physical actions do not have any sensing aspect, while sensing actions do not change the world. Part (c) simply says that $\mathcal{T}$ stops when no *do* symbols appear in s. We proceed now with the justifications for $\mathcal{R}$. Over equality atoms, $\mathcal{R}$ separates the terms of the equality and sends them to $\mathcal{T}$. Likewise, over non-fluent predicates. When *Poss* is encountered, preconditions take its place. Finally, $\mathcal{R}$ simplifies over connectives in a straightforward way. The main result for $\mathcal{R}$ regarding projection is:

Theorem 4 : *Suppose $\mathcal{D}$ is any action theory, ϕ any situation-suppressed formula and α any action sequence:*

$$\mathcal{D} \models \phi[do(\alpha, S_0)] \quad \textit{iff} \quad \mathcal{D}_0 \cup \mathcal{D}_{una} \models \mathcal{R}[\phi[do(\alpha, S_0)]]$$

where $\mathcal{D}_{una}$ is the unique name assumption and $\mathcal{R}[\phi[do(\alpha, S_0)]]$ mentions only a single situation term, S_0.

Here, $\mathcal{D}_{una}$ is only needed to simplify action terms (Reiter, 2001) *e.g.* from $fwd(4) = fwd(z)$, $\mathcal{D}_{una}$ infers $z = 4$. Now when our goal is to explicitly compute the degrees of belief in the sense of (7), we have the following property for $\mathcal{T}$:

Theorem 5: *Let $\mathcal{D}$ be as above, ϕ any situation-suppressed formula and α any sequence of actions. Then:*

$$\mathcal{D} \models Bel(\phi, do(\alpha, S_0)) = \mathcal{T}[Bel(\phi, do(\alpha, S_0))]$$

where $\mathcal{T}[Bel(\phi, do(\alpha, S_0))]$ is a term about S_0 only.

Theorem 5 essentially shows how belief about trajectories is computable using beliefs about S_0 only. Note that, since the result of $\mathcal{T}$ is a term about S_0, no sentence outside of $\mathcal{D} - \mathcal{D}_0$ is needed. We now illustrate regression with examples. Using Theorem 5, we reduce beliefs after actions to initial ones. At the final step, standard probabilistic reasoning is applied to obtain the end values.

Example 6: Let $\mathcal{D}$ contain the union of (2), (3) and (4).[11] Then the following equality expressions are entailed by $\mathcal{D}$:

1. $Bel(h = 10 \lor h = 11, S_0) = .2$

   $Bel(h \leq 9, S_0) = .8$

   Terms about S_0 are unaffected by $\mathcal{T}$. So this amounts to inferring probabilities using $\mathcal{D}_0$.

2. $Bel(h = 11, do(fwd(1), S_0))$

   $= \mathcal{T}[\ Bel(h = 11, do(fwd(1), S_0))\ ]$

   $= \mathcal{T}[\ Bel(\ \underline{\mathcal{R}[(h = 11)[do(fwd(1), now)]]\ , S_0})\ ]$ (i)

   $= \mathcal{T}[Bel(\ \underline{\mathcal{T}[h(do(fwd(1), now))] = \mathcal{T}[11]}\ , S_0)]$ (ii)

   $= \underline{\mathcal{T}[Bel(\max(0, h - 1) = 11, S_0)]}$ (iii)

   $= Bel(\max(0, h - 1) = 11, S_0)$ (iv)

   $= 0$

   First, since action preconditions are all true, *Poss* is ignored everywhere. We underline to emphasize the expressions undergoing transformations. We begin always by applying $\mathcal{T}$ to the main term, in this case getting (i), by means of $\mathcal{T}$'s item 4(a). Next, $\mathcal{R}$'s item 1 is applied in (ii). While $\mathcal{T}[11] = 11$ by $\mathcal{T}$'s item 1, for $\mathcal{T}[h(do(fwd(1), now))]$ we use item 3 and (2) to get:

   $$\mathcal{T}[\max(0, h(now) - 1)] = \max(0, h(now) - 1)$$

   which is substituted in (ii) to give (iii). Finally, $\mathcal{T}$'s item 4(c) yields (iv), which is a belief term about S_0. Now the only valid value for h in (iv) is 12, but for $h = 12$ the robot has a belief of 0 initially.

3. $Bel(h \leq 5, do(sonar(5), S_0))$

   $= \frac{1}{\gamma} \sum_{x \in \{2,\ldots,11\}} Err(5, x) \times \underline{\mathcal{T}[Bel(h = x \land h \leq 5, S_0)]}$ (i)

   $= \frac{1}{\gamma} \sum_{x \in \{2,\ldots,11\}} Err(5, x) \times Bel(h = x \land h \leq 5, S_0)$ (ii)

   $= \frac{1}{\gamma} \Big(\frac{1}{3} \cdot Bel(h = 4 \land h \leq 5, S_0)$
   $+ \frac{1}{3} \cdot Bel(h = 5 \land h \leq 5, S_0)$ (iii)
   $+ \frac{1}{3} \cdot Bel(h = 6 \land h \leq 5, S_0) \Big)$

   $= \frac{1}{\gamma} \Big(\frac{1}{3} \cdot Bel(h = 4, S_0) + \frac{1}{3} \cdot Bel(h = 5, S_0) \Big)$ (iv)

   $= \frac{1}{\gamma} \cdot \frac{2}{30}$

   $= 2/3$

   where $Err(5, x)$ is the model from (3). First, $\mathcal{T}$'s item 4(b) yields (i), and then item 4(c) yields (ii). Since $Err(5, x)$ is non-zero only for $x \in \{4, 5, 6\}$, (ii) is simplified to (iii) and (iv) resulting in $1/15 \cdot 1/\gamma$. We calculate γ as follows:

   $= \sum_{x \in \{2,\ldots,11\}} Err(5, x) \times \mathcal{T}[Bel(h = x \land true, S_0)]$ (i′)

   $= \sum_{x \in \{2,\ldots,11\}} Err(5, x) \times Bel(h = x, S_0)$ (ii′)

   $= 3/30.$

4 REGRESSION FOR GENERAL DOMAINS

We now generalize regression for domains with discrete and continuous variables, for which a *density*-based notion

[11] Initial beliefs can also be specified for $\mathcal{D}_0$ using *Bel*, *e.g.* (4) can be replaced in $\mathcal{D}_0$ with $Bel(h = u, S_0) = .1$ for $u \in \{2, \ldots, 11\}$.

of belief is appropriate. The main issue is that when formulating posterior beliefs after sensing, something like Definition 2's item 4(b) will not work. This is because over continuous spaces the belief about any individual point is 0. Therefore, we will be unpacking belief in terms of the density function, *i.e.* in terms of P. These $P(\vec{x}, \phi, s)$ terms, which will now also be treated as special sorts of syntactic terms, are separately regressed. (Of course, the regression of weight-based belief can be approached on similar lines.) Recall that $P(\vec{x}, \phi, S_0)$ is simply the *density* of an initial world (where $f_i = x_i$) satisfying ϕ. Formally, term regression $\mathcal{T}$ is defined as follows:

Definition 7: For any term t, we inductively define:

1, 2 and 3 as before.

4. $\mathcal{T}[P(\vec{x}, \phi, s)]$ is defined inductively:
   (a) if s is of the form $do(a, s')$ and a is a physical action then
   $$\mathcal{T}[P(\vec{x}, \phi, s)] = \mathcal{T}[P(\vec{x}, \psi, s')]$$
   where ψ is $Poss(a, now) \supset \mathcal{R}[\phi[do(a, now)]]$.
   (b) if s is of the form $do(a, s')$ and a is a sensing action $sense(z)$ such that $l(sense(z), s) = Err(z, f_i(s))$ is in $\mathcal{D}$, then:
   $$\mathcal{T}[P(\vec{x}, \phi, s)] = Err(z, x_i) \times \mathcal{T}[P(\vec{x}, \psi, s')]$$
   where ψ is $Poss(a, now) \supset \phi \wedge f_i(now) = x_i$.
   (c) else $\mathcal{T}[P(\vec{x}, \phi, s)] = P(\vec{x}, \phi, s)$.
5. $\mathcal{T}[Bel(\phi, s)] = \frac{1}{\gamma} \int_{\vec{z}} \sum_{\vec{y}} \mathcal{T}[P(\vec{z} \cdot \vec{y}, \phi, s)]$.

$\mathcal{R}$ is defined as before. We have the following property:

Theorem 8: *Let $\mathcal{D}$ be any action theory, ϕ any situation-suppressed formula and α any action sequence. Then*
$$\mathcal{D} \models Bel(\phi, do(\alpha, S_0)) = \mathcal{T}[Bel(\phi, do(\alpha, S_0))]$$
where $\mathcal{T}[Bel(\phi, do(\alpha, S_0))]$ is a term about S_0 only.

Similarly, the analogue of Theorem 4 holds as well.

Example 9: Consider the following *continuous* variant of the robot example. Imagine a continuous uniform distribution for the true value of h, as provided by (5). Suppose the sonar has the following error profile:
$$l(sonar(z), s) = \text{IF } z \geq 0 \text{ THEN } \mathcal{N}(z - h(s); 0, 4) \text{ ELSE } 0 \qquad (10)$$
which says the difference between a nonnegative reading and the true value is normally distributed with mean 0 and variance 4. (A mean of 0 implies there is no systematic bias.) Now, let $\mathcal{D}$ be any action theory that includes (2), (5) and (10). Then the following equalities are entailed by $\mathcal{D}$:

1. $Bel(h = 3 \vee h = 4, S_0) = 0$,
   $Bel(4 \leq h \leq 6, S_0) = .2$
   $\mathcal{T}$ does not change terms about S_0. Here, for example, the second belief term equals $\int_4^6 .1 dx = .2$.

2. $Bel(h \geq 11, do(fwd(1), S_0))$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \mathcal{T}[\, P(x, \underline{h \geq 11}, do(fwd(1), S_0)) \,] \qquad \text{(i)}$$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \mathcal{T}[P(x, \underline{\mathcal{R}[\psi]}, S_0)] \qquad \text{(ii)}$$
where ψ is $(h \geq 11)[do(fwd(1), now)]$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \underline{\mathcal{T}[P(x, \max(0, h-1) \geq 11, S_0)]} \qquad \text{(iii)}$$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} P(x, \max(0, h-1) \geq 11, S_0) \qquad \text{(iv)}$$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \begin{cases} p(\iota, S_0) & \text{if } \exists \iota.\, h(\iota) = x \wedge h(\iota) \geq 12 \\ 0 & \text{otherwise} \end{cases} \qquad \text{(v)}$$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \begin{cases} .1 & \text{if } x \in [2, 12] \text{ and } x \geq 12 \\ 0 & \text{otherwise} \end{cases} \qquad \text{(vi)}$$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \begin{cases} .1 & \text{if } x = 12 \\ 0 & \text{otherwise} \end{cases} \qquad \text{(vii)}$$
$$= 0$$
   We use $\mathcal{T}$'s item 5 to get (i), after which item 4(a) is applied. On doing $\mathcal{R}$ in (ii), along the lines of Example 6.2, we obtain (iii). $\mathcal{T}$'s item 4(c) then yields (iv), and stops. In the steps following (iv), we show how P expands in terms of p, and how the space of situations resolves into a mathematical expression, yielding 0.[12]

3. $Bel(h = 0, do(fwd(4), S_0))$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \mathcal{T}[P(x, \underline{\mathcal{R}[(h = 0)[do(fwd(4), now)]]}, S_0)] \qquad \text{(i)}$$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \mathcal{T}[P(x, \max(0, h-4) = 0, S_0)] \qquad \text{(ii)}$$
$$= \frac{1}{\gamma} \int_{x \in \mathbb{R}} \begin{cases} .1 & \text{if } x \in [2, 12] \text{ and } x \leq 4 \\ 0 & \text{otherwise} \end{cases} \qquad \text{(iii)}$$
$$= .2$$
   By means of (2), after moving forward by 4 units the belief about h is characterized by a *mixed* distribution because $h = 0$ is accorded a .2 weight (*i.e.* from all points where $h \in [2, 4]$ initially), while $h \in (0, 8]$ are associated with a density of .1. Here, $\mathcal{T}$'s item 5 and 4(a) are triggered, and the removal of $\mathcal{T}$ using 4(c) is not shown. The end result is that the density function is integrated for $2 \leq x \leq 4$ leading to .2. (γ is 1.)

[12]Given certain assumptions, it is possible to further reduce logical expressions involving fluents to a mathematical expression using only those variables that appear in the integral. We expand on this in a longer version of the paper.

4. $Bel(h = 4, do(fwd(-4), do(fwd(4), S_0)))$

$$= \frac{1}{\gamma}\int_{x\in\mathbb{R}} \mathcal{T}[P(x, \exists u.\ h = u \wedge 4 = \max(0, u + 4), do(fwd(4), S_0))] \quad \text{(i)}$$

$$= \frac{1}{\gamma}\int_{x\in\mathbb{R}} \mathcal{T}[P(x, \exists u.\ u = \max(0, h - 4) \ \wedge\ 4 = \max(0, u + 4), S_0)] \quad \text{(ii)}$$

$$= \frac{1}{\gamma}\int_{x\in\mathbb{R}} \begin{cases} .1 & \text{if } x \in [2, 12],\ x \le 4 \\ 0 & \text{otherwise} \end{cases} \quad \text{(iii)}$$

$= .2$

We noted above that the point $h = 4$ gets a .2 weight on executing *fwd*(4), after which it obtains a h value of 0. The weight is *retained* on reversing by 4 units, with the point now obtaining a h value of 4. The derivation invokes two applications of $\mathcal{T}$'s item 4(a). We skip the intermediate $\mathcal{R}$ steps. (γ evaluates to 1.)

5. $Bel(h = 4, do(fwd(4), do(fwd(-4), S_0)))$

$$= \frac{1}{\gamma}\int_{x\in\mathbb{R}} \mathcal{T}[P(x, \exists u.\ u = \max(0, h + 4) \ \wedge\ 4 = \max(0, u - 4), S_0)] \quad \text{(i)}$$

$= 0$

Had the robot moved away first, no "collapsing" of points takes place, h remains a continuous distribution and no point is accorded a non-zero weight. $\mathcal{T}$ steps are skipped but they are symmetric to the one above, *e.g.* compare (i) here and (ii) above. But then the density function is non-zero only for the individual $h = 4$.

6. $Bel(4 \le h \le 6, do(sonar(5), S_0))$

$$= \frac{1}{\gamma}\int_{x\in\mathbb{R}} \mathcal{N}(5 - x; 0, 4) \times \mathcal{T}[P(x, \psi, S_0)] \quad \text{(i)}$$

where ψ is $h = x \wedge 4 \le h \le 6$

$$= \frac{1}{\gamma}\int_{x\in\mathbb{R}} \begin{cases} .1 \cdot \mathcal{N}(5 - x; 0, 4) & \text{if } x \in [2, 12],\ x \in [4, 6] \\ 0 & \text{otherwise} \end{cases}$$

$\approx .41$

We obtain (i) after $\mathcal{T}$'s item 5 and then 4(b) for sensing actions. That is, belief about $h \in [4, 6]$ is sharpened after observing 5. Basically, we are integrating a function that is 0 everywhere except when $4 \le x \le 6$ where it is $.1 \times \mathcal{N}(5 - x; 0, 4)$, normalized over $2 \le x \le 12$.

7. $Bel(4 \le h \le 6, do(sonar(5), do(sonar(5), S_0))$

$$= \frac{1}{\gamma}\int_{x\in\mathbb{R}} \mathcal{N}(5 - x; 0, 4) \times \underline{\mathcal{T}[P(x, \psi, s))]} \quad \text{(i)}$$

where $s = do(sonar(5), S_0)$, ψ is $h = x \wedge 4 \le h \le 6$

$$= \frac{1}{\gamma}\int_{x\in\mathbb{R}} [\mathcal{N}(5 - x; 0, 4)]^2 \times \mathcal{T}[P(x, \psi, S_0)] \quad \text{(ii)}$$

$\approx .52$

As expected, two successive observations of 5 sharpens belief further. Derivations (i) and (ii) follow from $\mathcal{T}$'s item 5, and two successive applications of item 4(b). Thus, we are to integrate $.1\times[\mathcal{N}(5-x; 0, 4)]^2$ between $[4, 6]$ and normalize over $[2, 12]$. These changing densities are plotted in Figure 2.

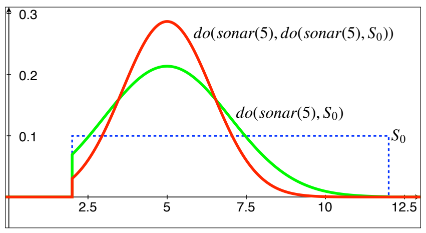


Figure 2: Belief density change for h at S_0 (in blue), after sensing 5 (in green) and after sensing 5 twice (in red).

5 TWO SPECIAL CASES

Regression is a general property for computing properties about posteriors in terms of priors after actions. It is therefore possible to explore limited cases, which might be appropriate for some applications. We present two such cases.

Conjugate distributions

Certain types of systems, such as Gaussian processes, admit an effective propagation model. The same advantages can be observed in our framework. We illustrate this using an example. Assume a fluent f, and suppose $\mathcal{D}_0$ is the union of (I), (P1) and the following specification:

$$p(\iota, S_0) = \mathcal{N}(f(\iota); \mu_1, {\sigma_1}^2)$$

which stipulates that the true value of f is believed to be normally distributed. Assume the following sensor in $\mathcal{D}$:

$$l(sense(z), s) = \mathcal{N}(z - f(s); \mu_2, {\sigma_2}^2)$$

Then it is easy to show that estimating posteriors yields a product of Gaussians (that is also a Gaussian process (Box and Tiao, 1973)), which is inferred by $\mathcal{T}$:[13]

$$\mathcal{T}[Bel(b \le f \le c, do(sense(z), S_0))] = \frac{1}{\gamma}\int_b^c \mathcal{N}(x; \mu_1, {\sigma_1}^2) \cdot \mathcal{N}(z - x; \mu_2, {\sigma_2}^2)\mathrm{d}x$$

Distribution transformations

Certain actions affect priors in a characteristically simple manner, and regression would account for these changes as an appropriate function of the initial belief state. We illustrate two instances using Example 9. First, consider an

[13]This corresponds to a simple case of Kalman filtering (Dean and Wellman, 1991), where the sensed value is static. In the complete framework with noisy effectors, we would obtain a model where distinct actions may condition priors in distinct ways.

action *grasp*(z) that grabs object z. Because the action of grasping does not affect h by way of (2), we get:

$$\mathcal{T}[Bel(h \leq b, do(grasp(obj5), S_0))] = Bel(h \leq b, S_0)$$

So no changes to h's density are required. Second, consider ground actions with the property that two distinct values of f do not become the same after that action, *e.g.*, for initial states this means:

$$\forall\iota, \iota'.\ f(\iota) \neq f(\iota') \supset f(do(a, \iota)) \neq f(do(a, \iota')) \qquad \text{(EQ)}$$

Think of *fwd*(-4) that agrees with this, but *fwd*(4) need not. We can show that such actions "shift" priors:

$$\mathcal{T}[Bel(h \leq b, do(fwd(-n), S_0))] = Bel(h \leq b - n, S_0)$$

Intuitively, the probability of h being in the interval $[b, c]$, irrespective of the distribution family, is the same as the probability of $h \in [b + n, c + n]$ after *fwd*($-n$). Thus, regression derives the initial interval given the current one.[14]

6 RELATED WORK

Perhaps the most popular models to treat sensor fusion include variants of Kalman filtering (Fox *et al.*, 2003; Thrun *et al.*, 2005), where priors and likelihoods are assumed to be Gaussian. We already pointed out some instances of Kalman filtering in our example. Where we differ is that backward chaining is possible even when: (a) no assumptions about the nature of distributions, nor about how distributions and dependencies change need to be made, (b) the framework is embedded in a rich theory of actions, and (c) arbitrary forms of incomplete knowledge are allowed, including strict uncertainty.[15] Domain-specific dependencies, then, may be exploited as appropriate.

There have been, of course, other attempts to extend the situation calculus to reason about probabilistic belief, such as (Poole, 1998). See BHL for a discussion on the differences to that work. On a related note, there are numerous approaches that combine logic and probability. In particular, we mention dynamic logic proposals (Van Benthem *et al.*, 2009), planning languages (Younes and Littman, 2004; Sanner, 2011; Kushmerick *et al.*, 1995), and first-order frameworks based on the situation calculus and close relatives (Thielscher, 2001). For discussions on these and non-dynamic proposals such as Markov logics (Richardson and Domingos, 2006), see (Belle and Levesque, 2013a,b).

There is one other thread of related work, that of *symbolic dynamic programming* (Boutilier *et al.*, 2000, 2001) which also has recent continuous extensions (Sanner *et al.*, 2011). While regression is used in this literature as well, the concerns are very different: they focus on policy generation, while ours is strictly about *belief change*. Consequently, the regression in that literature is adapted from the regression for the non-epistemic situation calculus (Reiter, 2001). Ours, on the other hand, continues in the tradition of the epistemic situation calculus (Scherl and Levesque, 2003) by extending those intuitions to probabilistic belief and noisy sensing. In this regard, our account allows the modeler to explicitly reason about beliefs in the language, which would prove useful in formalizing the achievability of plans (Levesque, 1996), among other things. The idea of regression is not new and lies at the heart of many planning systems (Fritz and McIlraith, 2007). For STRIPS actions, regression has at most linear complexity in the length of the action sequence (Reiter, 2001). For other studies, see (Van Ditmarsch *et al.*, 2007; Rintanen, 2008).

7 CONCLUSIONS

Planning and robotic applications have to deal with numerous sources of complexity regarding action and change. Consequently, irrespective of the decompositions and factorizations that are justifiable initially, belief state evolution is known to invalidate these efforts even over simple temporal phenomena. In this paper, we obtain a general methodology to relate beliefs after acting and sensing to initial beliefs. We investigated the methodology in an existing model by BHL, and a continuous extension to it, making the technique applicable to discrete domains as well as general ones. We demonstrated regression using an example where actions affect priors in nonstandard ways, such as transforming a continuous distribution to a mixed one. In general, regression does not insist on (but allows) restrictions to actions, that is, no assumptions need to be made about how actions affect variables and their dependencies over time. Moreover, at the specification level, we do not assume (but allow) structurally constrained initial states.

There are many avenues for future work. Extending automated regression solutions (Reiter, 2001) to subjective probabilities is ongoing work. Moreover, given the promising advances made in the area of relational probabilistic inference, we believe regression suggests natural ways to apply those developments with actions. This line of research would allow us to address effective belief propagation for numerous planning problems that require both logical and probabilistic representations. On another front, note that after applying the reductions, one may also use approximate inference methods. Perhaps then, regression can serve as a computational framework to study approximate belief propagation, on the one hand, and using approximate inference at the initial state after goal regression, on the other.

[14] Although *fwd* shifts the distribution linearly, in general, provided something like (EQ) is true, actions may also result in nonlinear changes to state variables, which would nonlinearly change the mean of the distribution. Nevertheless, similar features would be observed. See the longer version of the paper for more details.

[15] Moreover, the BHL model is compatible with a wide variety of formalisms such as (Fagin and Halpern, 1994; Halpern and Tuttle, 1993; Halpern, 1990). See BHL for discussions.

References

F. Bacchus, J. Y. Halpern, and H. J. Levesque. Reasoning about noisy sensors and effectors in the situation calculus. *Artificial Intelligence*, 111(1–2):171 – 208, 1999.

V. Belle and H. J. Levesque. Reasoning about continuous uncertainty in the situation calculus. In *Proc. IJCAI*, 2013.

V. Belle and H. J. Levesque. Robot location estimation in the situation calculus. In *ICAPS Workshop on Planning and Robotics*. 2013.

C. Boutilier, R. Reiter, M. Soutchanski, and S. Thrun. Decision-theoretic, high-level agent programming in the situation calculus. In *Proc. AAAI*, pages 355–362, 2000.

C. Boutilier, R. Reiter, and B. Price. Symbolic dynamic programming for first-order MDPs. In *Proc. IJCAI*, pages 690–697, 2001.

G. E. P. Box and G. C. Tiao. *Bayesian inference in statistical analysis*. Addison-Wesley, 1973.

X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proc. UAI*, pages 33–42, 1998.

T. Dean and K. Kanazawa. Probabilistic temporal reasoning. In *Proc. AAAI*, pages 524–529, 1988.

T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Computational intelligence*, 5(2):142–150, 1989.

T. Dean and M. Wellman. *Planning and control*. Morgan Kaufmann Publishers Inc., 1991.

R. Fagin and J. Y. Halpern. Reasoning about knowledge and probability. *J. ACM*, 41(2):340–367, 1994.

D. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello. Bayesian filtering for location estimation. *Pervasive Computing, IEEE*, 2(3):24–33, 2003.

C. Fritz and S. A. McIlraith. Monitoring plan optimality during execution. In *Proc. ICAPS*, pages 144–151, 2007.

V. Gogate and P. Domingos. Formula-based probabilistic inference. In *Proc. UAI*, pages 210–219, 2010.

H. Hajishirzi and E. Amir. Reasoning about deterministic actions with probabilistic prior and application to stochastic filtering. In *Proc. KR*, 2010.

J. Y. Halpern and M. R. Tuttle. Knowledge, probability, and adversaries. *J. ACM*, 40:917–960, 1993.

J.Y. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence*, 46(3):311–350, 1990.

N. Kushmerick, S. Hanks, and D.S. Weld. An algorithm for probabilistic planning. *Artificial Intelligence*, 76(1):239–286, 1995.

H. J. Levesque, F. Pirri, and R. Reiter. Foundations for the situation calculus. *Electron. Trans. Artif. Intell.*, 2:159–178, 1998.

H. J. Levesque. What is planning in the presence of sensing? In *Proc. AAAI / IAAI*, pages 1139–1146, 1996.

J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence*, pages 463–502, 1969.

J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

D. Poole. Decision theory, the situation calculus and conditional plans. *Electron. Trans. Artif. Intell.*, 2:105–158, 1998.

D. Poole. First-order probabilistic inference. In *Proc. IJCAI*, pages 985–991, 2003.

R. Reiter. *Knowledge in action: logical foundations for specifying and implementing dynamical systems*. MIT Press, 2001.

M. Richardson and P. Domingos. Markov logic networks. *Machine learning*, 62(1):107–136, 2006.

J. Rintanen. Regression for classical and nondeterministic planning. In *Proc. ECAI*, pages 568–572, 2008.

S. Sanner, K. V. Delgado, and L. N. de Barros. Symbolic dynamic programming for discrete and continuous state MDPs. In *Proc. UAI*, pages 643–652, 2011.

S. Sanner. Relational dynamic influence diagram language (rddl): Language description. Technical report, Australian National University, 2011.

R. B. Scherl and H. J. Levesque. Knowledge, action, and the frame problem. *Artificial Intelligence*, 144(1-2):1–39, 2003.

M. Thielscher. Planning with noisy actions (preliminary report). In *Proc. Australian Joint Conference on Artificial Intelligence*, pages 27–45, 2001.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

J. Van Benthem, J. Gerbrandy, and B. Kooi. Dynamic update with probabilities. *Studia Logica*, 93(1):67–96, 2009.

H. Van Ditmarsch, A. Herzig, and T. De Lima. Optimal regression for reasoning about knowledge and actions. In *Proc. AAAI*, pages 1070–1075, 2007.

R. Waldinger. Achieving several goals simultaneously. In *Machine Intelligence*, volume 8, pages 94–136. 1977.

H. Younes and M. Littman. PPDDL 1. 0: An extension to pddl for expressing planning domains with probabilistic effects. Technical report, Carnegie Mellon University, 2004.

Probabilistic Conditional Preference Networks

Damien Bigot
IRIT-CNRS
Toulouse University
31400 Toulouse, France
damien.bigot@irit.fr

Hélène Fargier
IRIT-CNRS
Toulouse University
31400 Toulouse, France
helene.fargier@irit.fr

Jérôme Mengin
IRIT-CNRS
Toulouse University
31400 Toulouse, France
jerome.mengin@irit.fr

Bruno Zanuttini
GREYC
Caen University
14000 Caen, France
bruno.zanuttini@unicaen.fr

Abstract

This paper proposes a "probabilistic" extension of conditional preference networks as a way to compactly represent a probability distributions over preference orderings. It studies the probabilistic counterparts of the main reasoning tasks, namely dominance testing and optimisation from the algorithmical and complexity viewpoints. Efficient algorithms for tree-structured probabilistic CP-nets are given. As a by-product we obtain a linear-time algorithm for dominance testing in standard, tree-structured CP-nets.

1 Introduction

Modelling preferences has been an active research topic in Artificial Intelligence for more than fifteen years. In recent years, several formalisms have been proposed that are rich enough to describe complex preferences of a user in a compact way, by e.g. Rao and Georgeff [1991], Gonzales *et al.* [2008], Boutilier *et al.* [2001, 2004]. Ordinal preferences, where alternatives, or outcomes, are ranked without the use of numerical functions, are usually easier to obtain, and are the topic of this paper.

In many contexts, the preferences of the user are ill-known, e.g. because they depend on the value of non controllable state variable, or because the system has no information about the user – her preferences may then be extrapolated from information gathered for previous customers. This is typically the case in anonymous recommendation systems, where several users with similar preferences can be grouped into a single model – that can then be finely tuned to fit one particular user. In this paper, we propose to use a probability distribution over preference models to represent ill known preferences. Specifically, we propose to extend conditional preference networks (CP-nets, one of the most popular ordinal preference representation formalisms [Boutilier *et al.*, 2004]) by attaching probabilities to the local preference rules.

Probabilistic CP-nets are evoked for preference elicitation in by de Amo *et al.* [2012]. However, the authors do not give a precise semantics to their CP-nets, nor do they study their computational properties. A more general form of Probabilistic CP-net is also described by Cornelio [2012], who prove that the problem of finding the most probably optimal outcome is similar to an optimisation problem in a Bayesian network.

In the present paper, we detail the Probabilistic CP model, and especially its semantics, and provide efficient algorithms to solve the corresponding dominance and optimisation problems. After a brief presentation of CP-nets (Section 2), we present Probabilistic CP-nets, their semantics and explain how they can be used in several practical settings (Section 3). In Section 4, we give efficient algorithms and complexity results for dominance testing. In Section 5, we turn to the optimisation task and prove that it can be performed in linear time when some restriction is put on the structure of the PCP-net. Section 6 concludes the paper.

2 Background

We consider combinatorial objects defined over a set of n variables $\mathcal{V}$. Variables are denoted by uppercase letters $A, B, X, Y, \ldots$. $\underline{X}$ denotes the domain of a variable X. More generally, for a set of variables $U \subseteq \mathcal{V}$, $\underline{U}$ denotes the Cartesian product of their domains. Elements of $\underline{\mathcal{V}}$ are called *objects* or *outcomes*, denoted by $o, o', \ldots$. Elements of $\underline{U}$ for some $U \subseteq \mathcal{V}$ are denoted by $u, u', \ldots$. Given two sets of variables $U \subseteq V \subseteq \mathcal{V}$ and $v \in \underline{V}$, we write $v[U]$ for the restriction of v to the variables in U.

In this paper we essentially consider variables with a Boolean domain. We consistently write x and $\bar{x}$ for the two values in the domain of X.

Preference Relations We assume that individual preferences can be represented by an order (reflexive, antisymmetric and transitive) over the set of all outcomes $\underline{\mathcal{V}}$. A convenient way to specify such orders over outcomes in a multi-attribute domain is by means of *local preference rules*: each rule enables one to compare outcomes that have some specific values for some attributes. Conditional preference networks [Boutilier *et al.*, 2004] enable direct comparisons between outcomes that differ in the value of one variable only (called *swap pairs* of outcomes). Such a rule has the form $(X, u\!:\!>)$, with $X \in \mathcal{V}$, $u \in \underline{U}$ for some $U \subseteq \mathcal{V} - \{X\}$, and $>$ a total order on $\underline{X}$. According to $(X, u\!:\!>)$, for every pair of outcomes o, o' such that $o[U] = o'[U] = u$ and $o[\mathcal{V} - (U \cup \{X\})] = o'[\mathcal{V} - (U \cup \{X\})]$, o is preferred to o' if and only if $o[X] > o[X']$. Intuitively, the rule $(X, u\!:\!>)$ can be read: "Whenever u is the case, outcomes are ordered as their values for X are ordered by $>$, everything else being equal".

Example 1. *Assuming a set of binary variables* $\mathcal{V} = \{X_1, \ldots, X_4\}$*, the rule* $(x_3, \bar{x}_2 : x_3 > \bar{x}_3)$ *entails that* $o = x_1\bar{x}_2x_3x_4$ *is preferred to* $o' = x_1\bar{x}_2\bar{x}_3x_4$*. On the other hand, it tells nothing about the preference between* o *and* $o'' = \bar{x}_1\bar{x}_2\bar{x}_3x_4$ *(everything else is* not *equal), nor between* $x_1x_2x_3x_4$ *and* $x_1x_2\bar{x}_3x_4$ *(it does not apply).*

Considering the transitive closure of the relation over swap pairs, the set of all outcomes can be (partially) ordered by a set R of such rules using the notion of *flip*. An *R-worsening flip* is an ordered swap pair (o, o') for which there is a rule $r = (X, u\!:\!>) \in R$ satisfying: $o[U] = o'[U] = u$, $o[\mathcal{V} \setminus (U \cup \{X\})] = o'[\mathcal{V} \setminus (U \cup \{X\})]$, and $o[X] > o'[X]$. A sequence of outcomes $o_1, \ldots, o_k$ is an *R-worsening sequence* if for $1 \leq i \leq k-1$, (o_i, o_{i+1}) is an R-worsening flip. We write $o \succ_R o'$ whenever there is an R-worsening sequence from o to o'. By construction, the relation $\succ_R$ precisely captures the transitive closure of the relation induced by R on swap pairs. We say that the set of rules R is *consistent* if $\succ_R$ is irreflexive, and *inconsistent* otherwise.

Conditional Preference Networks With a conditional preference network (CP-net), one can specify preferential dependencies between variables by means of a directed graph $G = (\mathcal{V}, E)$: an edge (X, Y) indicates that the preference over the domain of Y may depend on the values of X. Given such a graph and a vertex $X \in \mathcal{V}$, we write $\mathsf{pa}(X)$ for the set of *parents* of X in $(\mathcal{V}, E)$: $\mathsf{pa}(X) = \{Y \in \mathcal{V} \mid (Y, X) \in E\}$.

Definition 1 (CP-net)**.** *A (deterministic)* CP-net N *over a set of variables* $\mathcal{V}$ *is defined by a directed graph* $(\mathcal{V}, E)$*, and by a* conditional preference table *for each vertex / variable* $X \in \mathcal{V}$*, written* $\mathsf{CPT}(X)$*. The table* $\mathsf{CPT}(X)$ *gives a local preference rule* $(X, u\!:\!>)$ *for each combination of values* $u \in \underline{\mathsf{pa}(X)}$ *for the parents of* X*.*

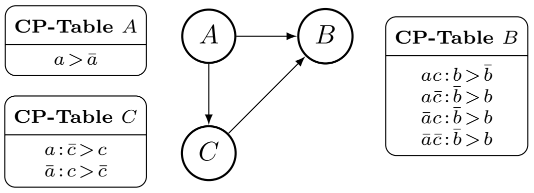


Figure 1: A deterministic CP-net

The graph G is called the structure *of N.*

When X is clear from the context, we write $u\!:\!>$ instead of $(X, u\!:\!>)$ for a conditional rule. For instance, given a CP-net and a binary variable B with a single parent A, we write $a\!:\!b > \bar{b}$ for the rule $(B, a\!:\!b > \bar{b})$. We also write $>^u_{N,X}$ for the total order over $\underline{X}$ specified by a CP-net N for some variable X and some assignment $u \in \underline{\mathsf{pa}(X)}$. Finally, if no ambiguity can arise, we use the same notation for a CP-net and its set of local preference rules. In particular, we write $o \succ_N o'$ to indicate that there is a worsening sequence from o to o' using the rules of N. When this is the case, we also say that N *entails* $o \succ o'$.

For complexity analysis, we write $|N|$ for the size of N, defined to be the number of symbols needed to write all rules, where writing a rule $(X, u\!:\!>)$ is considered to require $|U| + |\underline{X}|$ symbols. We also use specific classes of CP-nets, defined by restrictions on their structure G. For instance, the class of *acyclic* (resp. tree-structured) CP-nets is the class of CP-nets whose structure is an acyclic graph (resp. a forest).

A CP-net N is said to be *inconsistent* if the set of rules of N is inconsistent, and *consistent* otherwise. It is known [Boutilier *et al.*, 2004] that all acyclic CP-nets are consistent, but the converse is not true in general.

Example 2. *Figure 1 shows a CP-net over three variables* A, B, C*. This CP-net is consistent (it is acyclic), and it entails* $abc \succ \bar{a}\bar{b}c$*, as can be seen from the worsening sequence* $abc \succ a\bar{b}c \succ \bar{a}\bar{b}c$*, which uses the first rule in* $\mathsf{CPT}(B)$*, then the rule on* A*.*

Given a (consistent) CP-net N the two main reasoning problems are *dominance* and *optimisation*. Dominance is the problem of deciding whether N entails $o \succ o'$ for two given outcomes o, o', and optimisation consists in computing the "best" outcomes according to N; that is, the outcomes which are undominated under $\succ_N$. For acyclic CP-nets, optimisation is feasible in linear time, and there is always a unique optimal outcome. Contrastingly, testing dominance is PSPACE-complete for unrestricted CP-nets, NP-hard for acyclic ones, and quadratic for tree-structured ones [Goldsmith *et al.*, 2005].

3 Probabilistic CP-Nets

When the preferences of the user are ill-known, typically because they depend on the value of non controllable state variables, or because the system has few information about the user, we would like to be able to answer questions like "What is the probability that o is preferred to o' by some unknown agent?". *Probabilistic CP-nets* enable to compactly represent a probability distribution over CP-nets and answer such queries. A typical application is recommendation, where the preferences of the current (anonymous) user are extrapolated from profiles or from information gathered from previous customers in order to estimate how likely it is that a new customer makes a given choice.

Definition 2 (PCP-net). *A* probabilistic conditional preference network $\mathcal{N}$*, or* PCP-net*, over a set of variables $\mathcal{V}$, is defined by a directed graph $G=(\mathcal{V},E)$ and, for each vertex / variable $X\in\mathcal{V}$, a* probabilistic conditional preference table*, written* $\mathsf{PCPT}(X)$*. The PCP-table on X gives, for each assignment $u\in\underline{\mathsf{pa}(X)}$, a probability distribution over the set of the total orders on $\underline{X}$. We write $p^u_{\mathcal{N},X}$ for this distribution. We also call G the* structure *of $\mathcal{N}$.*

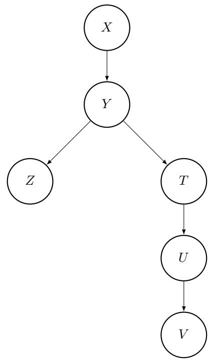
In particular, when all variables are binary, a PCP-table on X gives for each assignment $u\in\underline{\mathsf{pa}(X)}$ a probability distribution on the set of two orders $\{x>\bar{x},\bar{x}>x\}$[1] . For brevity, we write $u{:}x>\bar{x}$ (p) for the distribution which assigns probability p to $u{:}x>\bar{x}$ and $1-p$ to $u{:}\bar{x}>x$, as in Figure 2.

Taken as a whole, a PCP-net $\mathcal{N}$ is not intended to represent a preference relation. Rather, it represents a probability distribution over a set of (deterministic) CP-nets, namely, those which are compatible with $\mathcal{N}$.

Definition 3 (compatibility, probability). *A (deterministic) CP-net N is said to be* compatible *with a PCP-net $\mathcal{N}$, or to be $\mathcal{N}$-*compatible*, if it has the same structure as $\mathcal{N}$. In this case we write $N\propto\mathcal{N}$. If N is $\mathcal{N}$-compatible, we define the* probability *of N according to $\mathcal{N}$ by $p_{\mathcal{N}}(N)=\prod_{X\in V,u\in\underline{\mathsf{pa}(X)}}p^u_{\mathcal{N},X}(>^u_{N,X})$.*

It easily comes that $p_{\mathcal{N}}$ is a probability distribution over the set of deterministic $\mathcal{N}$-compatible CP-nets.

Example 3. *Figure 2 shows a PCP-net $\mathcal{N}$ over variables X,Y,Z,T,U,V. The first rule on Y, for instance, says that there is a .2 probability that a deterministic CP-net drawn at random contains the rule $x{:}y>\bar{y}$; otherwise (i.e. with probability $1-0.2$) it contains the opposite rule $x{:}\bar{y}>y$. Independently, there is a .3 probability that it contains $\bar{x}{:}y>\bar{y}$. In particular, there is a $.2\times.3=.06$ probability that it contains both and hence, that y is unconditionally preferred to $\bar{y}$.*



	$x>\bar{x}$	(.1)
x	$y>\bar{y}$	(.2)
$\bar{x}$	$y>\bar{y}$	(.3)
y	$z>\bar{z}$	(.5)
$\bar{y}$	$z>\bar{z}$	(.5)
y	$t>\bar{t}$	(.2)
$\bar{y}$	$t>\bar{t}$	(.7)
t	$u>\bar{u}$	(.1)
$\bar{t}$	$u>\bar{u}$	(.8)
u	$v>\bar{v}$	(.7)
$\bar{u}$	$v>\bar{v}$	(.6)

Figure 2: A probabilistic CP-net

The deterministic, $\mathcal{N}$-compatible CP-net with the negated value of each variable always preferred has probability $p=(1-.1)\times(1-.2)\times\cdots\times(1-.7)\times(1-.6)$.

Observe that when $\mathcal{N}$ contains cycles, $p_{\mathcal{N}}$ may be nonzero for some inconsistent CP-nets, which seems undesirable. Moreover , while deciding whether a given (cyclic) CP-net is consistent is a PSPACE-hard problem [Goldsmith *et al.*, 2005], the task is tractable in the acyclic case. Therefore, the remainder of the paper considers acyclic structures only.

Motivation Our motivation for studying PCP-nets stems from several different applicative settings. In the first one, the preference of the current (anonymous) user are unknown but the system has at its disposal the preferences of each of m individuals (e.g., past customers), and for each one the preferences are given by a (deterministic) CP-net N_i over some common structure G. Then the probabilistic CP-net $\mathcal{N}$ over the graph G defined by $p^u_{\mathcal{N},X}(>)=\#\{i\,|\,(X,u{:}>)\in N_i\}/m$ (proportion of N_i's which contains this rule, independently from other rules) provides a compact summary of the set of all individual preferences.

Such aggregation obviously induces a certain approximation of the distribution of preferences in the population. Namely, the probability of a given CP-net N as computed from the PCP-net $\mathcal{N}$ (Definition 3) is in general different from the proportion of individuals which indeed have the preferences encoded by N. Precisely, the construction amounts to approximate the distribution of preference relations as an independent one, considering each rule as a random variable. This may

[1]The situation might become less simple when the size of the domains increases: the PCP-table on X gives for each assignment $u\in\underline{\mathsf{pa}(X)}$ a probability distribution on the possible orders on $\underline{X}$. Allthrough there are $|\underline{X}|!$ potential orders, only a few may receive a significant probability and shall explicity appear in the PCP table; the remaining mass of probabilty is then assumed to be shared by the other, less significant, preference orders.

look like a crude approximation; still, as shown below, it is sound and complete for some restricted queries. Moreover, we discuss in Section 6 how PCP-nets can be extended to richer representations of distributions.

A close setting in which PCP-nets may prove useful is the one where a system interacts with a lot of individuals, but each one gives only a few preferences. For instance, in a recommender system, assume that each customer implicitly gives a preference of the form $u : x > \bar{x}$, by choosing one of two objects in a swap pair. This is the case when, say, a customer chooses the colour for a car in a context of interactive configuration [Gelle and Weigel, 1996]: she implicitly expresses a preference of the form $u : x > \bar{x}$, where U is the set of variables that have already been assigned and x is the chosen colour. Individual (deterministic) rules are thus obtained from different customers and, in the absence of other information, it clearly makes sense to aggregate these rules independently from each other.

A third applicative context is one in which only one person or agent expresses her preferences, but some noise must be taken into account, due to the elicitation process, or possibly from the person's preferences themselves (*e.g.*, "for dinner, with pasta bolognese I *most often* prefer having parmiggiano"). Assuming independent noise on each rule, PCP-nets are well suited for representing such preferences (through a rule like: $\text{dinner} \wedge \text{bolognese} : \text{parmiggiano} > \neg\text{parmiggiano} (.9)$ for the above example).

In all these settings, a PCP-net comes with a structure, which constrains the dependencies among variables. In case several CP-nets are aggregated into a PCP-net, it is natural to build the latter with the union of all individual graphs as its own structure. Indeed, an individual CP-net with structure (V, E) can always be seen as one over (V, E'), for any superset E' of E. In the remainder of this paper we will mainly focus on *tree-structured* (P)CP-nets. While this is a clear restriction on expressivity, as we will see even such networks raise nontrivial computational problems.

Reasoning Tasks Since a PCP-net represents a probability distribution on a set of deterministic CP-nets, the most natural queries are the following.

Definition 4 (probability of dominance). *Given a PCP-net $\mathcal{N}$ and two outcomes o, o', the* probability of $o \succ o'$, *written $p_\mathcal{N}(o \succ o')$, is defined to be the probability mass of $\mathcal{N}$-compatible CP-nets which entail $o \succ o'$:*

$$p_\mathcal{N}(o \succ o') = \sum_{N \propto \mathcal{N}, o \succ_N o'} p_\mathcal{N}(N)$$

Clearly enough, the probability of $o \succ o'$ given $\mathcal{N}$ is precisely the probability, when drawing a CP-net at random according to $p_\mathcal{N}$, of obtaining one which entails $o \succ o'$. In the remainder of the paper, we will essentially study how to compute such probability.

The second query is the probabilistic counterpart of optimisation in deterministic CP-nets.

Definition 5 (probability of being optimal). *Given an acyclic PCP-net $\mathcal{N}$ and an outcome o, the* probability for o to be optimal, *written $p_\mathcal{N}(o)$, is defined to be the probability mass of $\mathcal{N}$-compatible CP-nets which have o as their optimal outcome* [2].

Interestingly, despite the important approximation induced when summarising a population of CP-nets into a single PCP-net, some reasoning tasks can be performed exactly with the approximation (PCP-net) only. So let $\mathcal{N}$ be an acyclic PCP-net built from the rulewise aggregation of individual CP-nets.

Proposition 1. *Let $\mathcal{N}$ be an acyclic PCP-net and $\{o, o'\}$ a swap pair of outcomes, differing only on the value of X. The probability $p_\mathcal{N}(o \succ o')$ is precisely the proportion of individual CP-nets which entail $o \succ o'$.*

Proof. This follows from the fact that for acyclic G, a deterministic CP-net N entails $o \succ o'$ if and only if it contains the rule $o[\mathsf{pa}(X)] : o[X] > \bar{o}[X]$ [Koriche and Zanuttini, 2009, Lemma 1]. $\square$

Another interesting property is the preservation of local Condorcet winners [Xia *et al.*, 2008, Li *et al.*, 2011], also called "hypercubewise Condorcet winners" by Conitzer *et al.* [2011]: they are the outcomes o which are preferred by at least one half of the individual CP-nets to all o' that differ from o in the value of one variable only. Proposition 1 proves that the hypercubewise Condorcet winners are the outcomes that dominate each of their neighbors in the aggregated PCP-net with a probability of at least 0.5.

Moreover, let us insist that PCP-nets may serve other purposes than preference aggregation, as, for instance, modelling ill-known preferences of a single user, and that in such settings no approximation occurs.

4 Complexity of Dominance Testing

We now study the complexity of the dominance problem, namely, of computing the probability of $o \succ o'$ given a PCP-net $\mathcal{N}$. We restrict our attention to tree-structured CP-nets, that is, to the case when G is acyclic and assigns at most one parent to each variable. This arguably cannot capture all interesting dependency structures among variables, but as we will see this is already a nontrivial setting.

[2]Under our assumption of acyclicity, each CP-net is guaranteed to have a unique optimal outcome, hence the soundness of the definition.

We first give a generic construction, and use it for deriving a fixed-parameter tractability result, with the number of variables over which o, o' differ as the parameter. Then as a by-product, we derive an interesting result for *deterministic* CP-nets, namely, an $O(n)$ algorithm for dominance testing. Finally, we show that with slightly more general structures, computing the probability of dominance is #P-hard.

4.1 Construction

The cornerstone of our results is a characterisation of all deterministic CP-nets for which there exists a worsening sequence from o to o', given a tree structure G. The characterisation is given as a propositional formula for each leaf X, written $\mathsf{worsen}^{o,o'}(X)$, over variables of the form $y\!:\!x > \bar{x}$, $\bar{y}\!:\!x > \bar{x}$, etc., with Y the parent of X in G. An assignment of, say, $y\!:\!x > \bar{x}$ to $\top$, means that the corresponding CP-net contains the rule; a complete assignment to all variables thus defines a deterministic CP-net with structure G.

Precisely, fix a forest $G = (V, E)$ and two outcomes o, o'. For each variable X with no parent in G, we introduce the propositional variable $x > \bar{x}$, and we write $\bar{x} > x$ for its negation (because $>$ is total, $x > \bar{x}$ is true iff $\bar{x} > x$ is false) . Similarly, for each variable X with $\mathsf{pa}(X) = \{Y\}$, and $y^{\epsilon} \in \{y, \bar{y}\}$, we introduce the propositional variables $y\!:\!x > \bar{x}$ and $\bar{y}\!:\!x > \bar{x}$ and we write $y\!:\!\bar{x} > x$ and $\bar{y}\!:\!\bar{x} > x$ for their respective negations.

Boutilier *et al.* [2004, Appendix A] show that a worsening sequence may include up to $\Theta(n)$ changes of the value of some variable, even with binary tree-structured CP-nets. We exploit it by reasoning on the number of changes of each variable.

Precisely, the formula $\mathsf{change}_k^{o,o'}(X)$ means that there is a worsening sequence in which X alternates value at least k times, starting from its value in o and ending with its value in o'. For instance, $\mathsf{change}_3^{x\cdots,\bar{x}\cdots}(X)$ means that there is a worsening sequence in which X successively takes values $x, \bar{x}, x, \bar{x}$ (at least 4 values and 3 alternations). Formula $\mathsf{change}_k^{o,o'}(X)$ is defined inductively in Table 1, where Y denotes the parent of X. We give the formulas for the case where $o[X] = x, o[Y] = y$, the other cases can be obtained by symetry. Then $\mathsf{worsen}^{o,o'}(X)$ is defined as follows:

- $\mathsf{worsen}^{o,o'}(X) = \mathsf{change}_0^{o,o'}(X)$ if $o[X] = o'[X]$;
- $\mathsf{worsen}^{o,o'}(X) = \mathsf{change}_1^{o,o'}(X)$ otherwise.

Example 4. *Consider again the PCP-net depicted in Figure 2, and let $o = xyztuv, o' = \bar{x}y\bar{z}t\bar{u}v$. The corresponding formulas are given in Table 2.*

In the following, we write $o[\geq X]$ for o restricted to the variables which are ascendants of X in G (X included).

Proposition 2. *There is a worsening sequence from $o[\geq X]$ to $o'[\geq X]$ in which X changes value at least k times if and only if N is a model of $\mathsf{change}_k^{o,o'}(X)$.*

Proof. The proof goes by induction on the definition of the formula. For lack of space, we omit the proof for the base cases.

For the inductive step, we give a proof only for Rule 1 (Case $o[X] = o'[X] = x, o[Y] = o'[Y] = y$). The other rules are proved in exactly the same manner. So assume first that N satisfies the formula in Rule 1. Then by IH there is a worsening sequence

$$\omega_1 y, \omega_2 \bar{y}, \ldots, \omega_k \bar{y}, \omega_{k+1} y$$

in which all ω_i's are assignments to the proper ascendants of Y and $\omega_1 y$ (resp. $\omega_{k+1} y$) is $o[\geq Y]$ (resp. $o'[\geq Y]$). If moreover N satisfies the first disjunct ($y\!:\!x > \bar{x} \wedge \bar{y}\!:\!\bar{x} > x$), since the value of X has no influence on the preference over the values of Y we can build the sequence

$$\omega_1 yx, \omega_1 y\bar{x}, \omega_2 \bar{y}\bar{x}, \omega_2 \bar{y}x, \ldots, \omega_k \bar{y}\bar{x}, \omega_k \bar{y}x, \omega_{k+1} yx$$

which is a worsening sequence from $o[\geq X]$ to $o'[\geq X]$ where X changes value k times, as desired. Similarly, if N satisfies the second disjunct, we can build the following sequence, where X also changes value k times.

$$\omega_1 yx, \omega_2 \bar{y}x, \omega_2 \bar{y}\bar{x}, \ldots, \omega_k \bar{y}x, \omega_k \bar{y}\bar{x}, \omega_{k+1} y\bar{x}, \omega_{k+1} yx$$

Conversely, we show that if there is a sequence as in the claim, then N satisfies the formula in Rule 1. Let

$$\omega_1 x, \omega_2 \bar{x}, \ldots, \omega_k \bar{x}, \omega_{k+1} x$$

be a sequence from $o[\geq X]$ to $o'[\geq X]$ in which x changes value at least $k \geq 2$ times. There must be two opposite rules on X, for otherwise X cannot change value back and forth. Hence the disjunction in the definition of $\mathsf{change}_k^{o,o'}(X)$ is satisfied. Moreover, these rules must fire alternatively at least k times overall, hence Y must take at least k different values in the sequence $\omega_1, \omega_2, \ldots, \omega_{k+1}$, that is, change value at least $k-1$ times. But since it starts and ends with the same value y and $k-1$ is odd, in fact it must change at least k times. Hence by IH, N must satisfy $\mathsf{change}_k^{o,o'}(Y)$. □

Proposition 3. *There is a worsening sequence from o to o' if and only if N satisfies the formula $\bigwedge_X \mathsf{worsen}^{o,o'}(X)$, where X ranges over all leaves in the tree structure of N.*

Proof. Proposition 2 shows the claim if G is reduced to a chain. For the more general setting, consider two branches with a common part above X (included), and

Base cases ($\mathbf{Pa(X)}=\emptyset$ or $\mathbf{k}\leq\mathbf{1}$)			
$\mathbf{Pa(X)}$	$\mathbf{o}, \mathbf{o'}$	$\mathbf{k}$	$\mathsf{change}_{\mathbf{k}}^{\mathbf{o},\mathbf{o'}}(\mathbf{X})$
$\emptyset$	$o[X]=o'[X]$	0	$\top$
$\emptyset$	$o[X]=o'[X]$	>0	$\bot$
$\emptyset$	$o[X]=x, o'[X]=\bar{x}$	0	$\mathsf{change}_1^{o,o'}(X)$
$\emptyset$	$o[X]=x, o'[X]=\bar{x}$	1	$x>\bar{x}$
$\emptyset$	$o[X]=x, o'[X]=\bar{x}$	>1	$\bot$
$\{Y\}$	$o[X]=o'[X]$	0	$\mathsf{change}_0^{o,o'}(Y)$
$\{Y\}$	$o[X]\neq o'[X]$	0	$\mathsf{change}_1^{o,o'}(X)$
$\{Y\}$	$o[X]=o'[X]$	1	$\mathsf{change}_2^{o,o'}(X)$
$\{Y\}$	$o[X]=x, o'[X]=\bar{x}, o[Y]=o'[Y]=y$	1	$(y:x>\bar{x}\wedge\mathsf{change}_0^{o,o'}(Y)) \vee (\bar{y}:x>\bar{x}\wedge\mathsf{change}_2^{o,o'}(Y))$
$\{Y\}$	$o[X]=x, o'[X]=\bar{x}, o[Y]=y, o'[Y]=\bar{y}$	1	$(y:x>\bar{x}\vee\bar{y}:x>\bar{x}) \wedge \mathsf{change}_1^{o,o'}(Y)$

Inductive step ($\mathbf{Pa(X)}\neq\emptyset$ and $\mathbf{k}>\mathbf{1}$)			
Rule	$\mathbf{k}$	$\mathbf{o[X]}, \mathbf{o'[X]}, \mathbf{o[Y]}, \mathbf{o'[Y]}$	$\mathsf{change}_{\mathbf{k}}^{\mathbf{o},\mathbf{o'}}(\mathbf{X})$
0	odd	$x, x, \text{indifferent}, \text{indiff.}$	$\mathsf{change}_{k+1}^{o,o'}(X)$
1	even	x, x, y, y	$((y:x>\bar{x}\wedge\bar{y}:\bar{x}>x)\vee(y:\bar{x}>x\wedge\bar{y}:x>\bar{x})) \wedge \mathsf{change}_k^{o,o'}(Y)$
2	even	$x, x, y, \bar{y}$	$(y:x>\bar{x}\wedge\bar{y}:\bar{x}>x\wedge\mathsf{change}_{k-1}^{o,o'}(Y)) \vee (y:\bar{x}>x\wedge\bar{y}:x>\bar{x}\wedge\mathsf{change}_{k+1}^{o,o'}(Y))$
3	even	$x, \bar{x}, \text{indifferent}, \text{indiff.}$	$\mathsf{change}_{k+1}^{o,o'}(X)$
4	odd	$x, \bar{x}, y, y$	$(y:x>\bar{x}\wedge\bar{y}:\bar{x}>x\wedge\mathsf{change}_{k-1}^{o,o'}(Y)) \vee (y:\bar{x}>x\wedge\bar{y}:x>\bar{x}\wedge\mathsf{change}_{k+1}^{o,o'}(Y))$
5	odd	$x, \bar{x}, y, \bar{y}$	$((y:x>\bar{x}\wedge\bar{y}:\bar{x}>x)\vee(y:\bar{x}>x\wedge\bar{y}:x>\bar{x})) \wedge \mathsf{change}_k^{o,o'}(Y)$

Table 1: Inductive definition of the formula $\mathsf{change}_k^{o,o'}(X)$

$$\begin{aligned}
\mathsf{worsen}^{o,o'}(V) &= \mathsf{worsen}^{o,o'}(U)\\
\mathsf{worsen}^{o,o'}(U) &= (t:u>\bar{u}\wedge\mathsf{change}_0^{o,o'}(T)) \vee (\bar{t}:u>\bar{u}\wedge\mathsf{change}_2^{o,o'}(T))\\
\mathsf{change}_0^{o,o'}(T) &= \mathsf{worsen}^{o,o'}(Y)\\
\mathsf{change}_2^{o,o'}(T) &= ((y:t>\bar{t}\wedge\bar{y}:\bar{t}>t)\vee(y:\bar{t}>t\wedge\bar{y}:t>\bar{t})) \wedge \mathsf{change}_2^{o,o'}(Y)\\
\mathsf{worsen}^{o,o'}(Z) &= (y:z>\bar{z}\wedge\mathsf{change}_0^{o,o'}(Y)) \vee (\bar{y}:z>\bar{z}\wedge\mathsf{change}_2^{o,o'}(Y))\\
\mathsf{worsen}^{o,o'}(Y) &= \mathsf{change}_0^{o,o'}(Y)\\
\mathsf{change}_0^{o,o'}(Y) &= \mathsf{worsen}^{o,o'}(X)\\
\mathsf{change}_2^{o,o'}(Y) &= (x:y>\bar{y}\wedge\bar{x}:\bar{y}>y\wedge\mathsf{change}_1^{o,o'}(X)) \vee (x:\bar{y}>y\wedge\bar{x}:y>\bar{y}\wedge\mathsf{change}_3^{o,o'}(X))\\
\mathsf{change}_1^{o,o'}(X) &= x>\bar{x}\\
\mathsf{change}_3^{o,o'}(X) &= \bot
\end{aligned}$$

Table 2: Formulas for the example of Figure 2 with $o=xyztuv, o'=\bar{x}y\bar{z}t\bar{u}v$

write $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ for the set of variables above X (included), in the left subtree below X, and in the right subtree below X, respectively.

Clearly, if there is a worsening sequence from o to o', then N must satisfy the formula for both branches (by Proposition 2). For the converse, if N satisfies both formulas, by Proposition 2 again there is a worsening sequence from the outcome $o[\geq Y]=o[\mathcal{X}]o[\mathcal{Y}]$ to $o'[\geq Y]=o'[\mathcal{X}]o'[\mathcal{Y}]$, and one from $o[\mathcal{X}]o[\mathcal{Z}]$ to $o'[\mathcal{X}]o'[\mathcal{Z}]$. By construction of the formula $\mathsf{worsen}^{o,o'}(\cdot)$, there is one of these sequences in which the values of the variables above $\mathcal{X}$ change most, say, the one for $\mathcal{Y}$. Then since $\mathcal{Y}$ and $\mathcal{Z}$ are independent of each other, all flips over $\mathcal{Z}$ can also be performed in this sequence and interleaved with those over $\mathcal{Y}$. In this manner we get a worsening sequence from o to o', as desired.

The proof for a generic forest is obtained by applying this reasoning inductively on the set of branches. $\square$

4.2 Efficient Dominance Testing

From Propositions 2–3 we first derive a *fixed-parameter tractable* (FPT) algorithm for dominance testing in tree-structured PCP-nets. Recall that a FPT algorithm is one with running time $O(f(k).n^c)$, where n is the size of the input, c is a constant, f is a computable function, and k is some measure of the input size, called the *parameter* and assumed to be small [Flum and Grohe, 2006]. The running time of such an algorithm is essentially a polynomial modulo a factor which may be exponential (or more) in the value of the parameter.

As a parameter for the dominance problem in PCP-nets, we take the number of variables which have a different value in o and o'. This makes sense in practice since typically, in applications, one does not have to compare objects which are completely different from each other. For instance, in recommender systems a recommendation is likely to take place once the customer has fixed a number of features of the product which she wants to buy (*e.g.*, "I want a recent Blues album, cheaper than such price, etc.").

Definition 6. *The* parameterized dominance problem for tree-structured PCP-nets*, written* `p-Tree-PDominance`*, is defined by:*

Input:	*a tree-structured PCP-net $\mathcal{N}$, o, o'*
Parameter:	$k = \lvert\{X \in \mathcal{V} \mid o[X] \neq o'[X]\}\rvert$
Output:	*the probability of $o \succ o'$ according to $\mathcal{N}$*

Theorem 1. *The problem* `p-Tree-Dominance` *is fixed-parameter tractable. Precisely, it admits an algorithm with running time in $O(2^{2k^2} n)$.*

Proof. For each leaf variable X in the tree of $\mathcal{N}$, the algorithm first unrolls the formula $\mathsf{worsen}^{o,o'}(X)$. Each time if finds two different recursive calls (*e.g.*, on $k-1$ and $k+1$ in the second rule), it splits the formula into two parts. By construction the algorithm ends up with

$$\Phi^X = \{\varphi_1^X, \varphi_2^X, \ldots, \varphi_{n_X}^X\}.$$

The φ_i^X are mutually inconsistent, since the recursive calls in each rule are conditioned on mutually inconsistent formulas about the current node. Moreover, by Proposition 2, a CP-net $N \propto \mathcal{N}$ satisfies $o[\geq X] \succ o'[\geq X]$ if and only if it satisfies one of these formulas.

Now define Φ to be the set of formulas

$$\Phi = \{\varphi^{X_1} \wedge \varphi^{X_2} \wedge \cdots \wedge \varphi^{X_k} \mid X_i \text{ a leaf}, \varphi^{X_i} \in \Phi^{X_i}\}$$

that is, the "cartesian products" of the Φ^X's (over all leaves). By construction, the conjunctions in Φ are mutually inconsistent, and a CP-net $N \propto \mathcal{N}$ satisfies $o \succ o'$ if and only if it satisfies one of them (Proposition 3). It follows that the probability sought for can be computed in time $O(|\Phi|\ .\ n)$: the weight of each conjunction of Φ can be obtained by multiplying the probabilities of the corresponding rules in $\mathcal{N}$, in time $O(n)$, and by mutual inconsistency the result is obtained by summing up over the elements of Φ. Observe that some elements of Φ may be inconsistent formulas, but this can be detected efficiently since by construction they are conjunctions of literals.

To complete the proof we only need to bound the size of Φ. First consider Φ^X for some variable X: by construction, $|\Phi^X|$ is 2^ℓ, where ℓ is the number of rules used which result in two different recursive calls. This is the case for the second and third rules of Table 1 only, that is, when exactly one of X, Y has a different value in o and o'. It follows $\ell \leq 2k$, hence $|\Phi^X| \leq 2^{2k}$ for all X and finally, $|\Phi| \leq (2^{2k})^k = 2^{2k^2}$, as claimed. $\square$

4.3 The Deterministic Case

As an interesting by-product, we now derive a linear-time algorithm for dominance testing in tree-structured *deterministic* CP-nets. This improves on the quadratic running time of the TreeDT algorithm of Boutilier *et al.* [2004], and may seem odd since the smallest worsening sequence may be of quadratic size. This is actually not contradictory: our result says that it is possible to decide whether this sequence exists, without explicitly constructing it.

Theorem 2. *The dominance problem for tree-structured (deterministic) CP-nets on n variables can be solved in linear time $O(n)$.*

Proof. The algorithm[3] simply consists of deciding whether N satisfies the formula $\bigwedge_X \mathsf{worsen}^{o,o'}(X)$, where X ranges over all leaves in the structure of N. This can be done efficiently because for all four general rules, N necessarily satisfies at most one of the two disjuncts and hence, only one recursive call is involved at each step. The only point to be checked is that the algorithm can avoid considering the same variable several times along different branches.

To do so, the algorithm unrolls the formulas $\mathsf{worsen}^{o,o'}(X)$ in parallel. Each time two branches meet at a node X, this must be through recursive calls fired by the children of X. By construction, these calls

[3]This proof is a direct application of our PCP-algorithm to the deterministic case; but the query may be addressed by more dedicated and simpler (linear) algorithms. We thank an anonymous reviewer for pointing this to us.

are all of the form $\mathsf{change}_{k_i}^{o,o'}(X)$, and by construction and Proposition 3, all of them must be satisfied.

Recall that $\mathsf{change}_{k_i}^{o,o'}(X)$ reads "X changes value *at least* k_i times". Then the algorithm simply needs to replace all recursive calls by a unique one, namely, $\mathsf{change}_{\max_i(k_i)}^{o,o'}(X)$. In the end each variable is visited once, and the algorithm is indeed linear-time. $\square$

Interestingly, a top-down algorithm is also possible: starting from the root nodes in the structure of N, inductively computes for each node X the greatest value k such that N satisfies the formula $\mathsf{change}_{k}^{o,o'}(X)$. This algorithm allows us to derive the following result about *incomplete* deterministic CP-nets.

Say that a deterministic CP-net N is *incomplete with a given structure* if it comes with a graph G but for some variables X and assignments u to their parents, N contains neither the rule $u\!:\!x>\bar{x}$ nor the opposite rule $u\!:\!\bar{x}>x$. Incomplete CP-nets arise naturally in the process of elicitation [Koriche and Zanuttini, 2009], and more generally when a user is indifferent to some objects (for instance: "I have no preferred colour for motorbikes, since I don't like motorbikes at all"). Then a *completion* of N is a (complete, deterministic) CP-net with structure G and containing the rules of N.

Theorem 3. *The problem of deciding whether there is at least one completion of a given, incomplete CP-net N with a given tree structure, which entails $o>o'$ for given o, o', can be solved in linear time $O(n)$.*

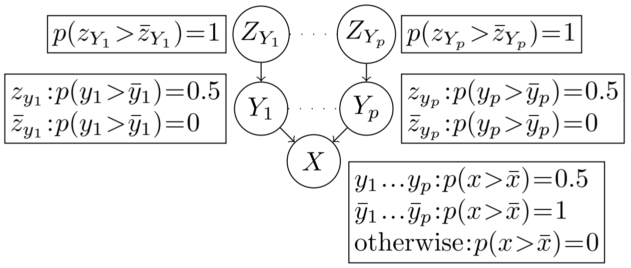
Proof. As evoked above, proceed top-down in the tree, by computing for each node X the greatest k for which there is a completion of N satisfying $\mathsf{change}_{k}^{o,o'}(X)$. To do so, complete all missing rules in a greedy manner. For instance, if the current node Y and its child X are in the setting of Inductive Step 2 of Table 1, and N contains no rule over X, choose the rules in the first disjunct to add in the completion of N. In this manner, from the value k for Y we get $k+1$ for X.

Obviously (because $\mathsf{change}_{k}^{o,o'}(X)$ reads "at least k times"), the greater the value k at each node, the more chances there are that the current completion indeed entails $o\succ o'$, hence the algorithm is correct. $\square$

4.4 Hardness Result

We conclude this section by giving a hardness result, which sheds light on the difficulty of testing dominance in PCP-nets with a more general structure than a tree.

Theorem 4. *The problem of computing $p_{\mathcal{N}}(o\succ o')$, given a PCP-net $\mathcal{N}$ and two outcomes o and o', is #P-hard. This holds even if the structure is acyclic, the longest path has length 3, each node has at most one outgoing edge and at most 4 parents.*



$p(z_{Y_1}>\bar{z}_{Y_1})=1$
Z_{Y_1}
Z_{Y_p}
$p(z_{Y_p}>\bar{z}_{Y_p})=1$
$z_{y_1}\!:\!p(y_1>\bar{y}_1)=0.5$
$\bar{z}_{y_1}\!:\!p(y_1>\bar{y}_1)=0$
Y_1
Y_p
$z_{y_p}\!:\!p(y_p>\bar{y}_p)=0.5$
$\bar{z}_{y_p}\!:\!p(y_p>\bar{y}_p)=0$
X
$y_1\ldots y_p\!:\!p(x>\bar{x})=0.5$
$\bar{y}_1\ldots\bar{y}_p\!:\!p(x>\bar{x})=1$
otherwise$\!:\!p(x>\bar{x})=0$

Figure 3: Reduction scheme

Proof. We give a reduction from #Monotone (2-4μ) Bipartite CNF, which is #P-complete [Vadhan, 2002].

Let $\mathcal{X}$ and $\mathcal{Y}$ be two disjoint sets of variables. A monotone (2-4μ)-bipartite CNF is a conjunction of clauses of the form $X\vee Y$, with $X\in\mathcal{X}$ and $Y\in\mathcal{Y}$, such that no variable appears more than 4 times in the formula. Given such a formula ϕ, we build a PCP-net $\mathcal{N}$ over $\mathcal{V}=\mathcal{X}\cup\mathcal{Y}\cup\mathcal{Z}$, where $\mathcal{Z}$ contains one fresh variable, written Z_Y, for each $Y\in\mathcal{Y}$. The variables of $\mathcal{Z}$ have no parent, each $Y\in\mathcal{Y}$ has a single parent Z_Y, and each $X\in\mathcal{X}$ has for parents the Y's such that the clause $X\vee Y$ appears in ϕ (there are at most 4 of them). This structure and the probability of each rule are given in Figure 3, where we show the portion of the PCP-net that corresponds to clauses $X\vee Y_1, \ldots, X\vee Y_p$.

Now consider the two outcomes o, o' defined by $o[X]=x, o'[X]=\bar{x}$ for every $X\in\mathcal{X}$, $o[Y]=o'[Y]=y$ for every $Y\in\mathcal{Y}$, and $o[Z]=z, o'[Z]=\bar{z}$ for every $Z\in\mathcal{Z}$. We show that $p_{\mathcal{N}}(o\succ o')$ is exactly the proportion of interpretations of $\mathcal{V}$ in which ϕ is true.

Let I be an interpretation of $\mathcal{X}\cup\mathcal{Y}$, and define the deterministic CP-net $N_I\propto\mathcal{N}$ as follows:

(1) for every $\mathbf{Z}\in\mathcal{Z}$, N_I contains $z>\bar{z}$; and

(2) for every $\mathbf{Y}\in\mathcal{Y}$: (a) N_I contains $\bar{z}_Y\!:\!\bar{y}>y$, and (b) if $I(Y)=\top$ then N_I contains $z_Y\!:\!y>\bar{y}$, otherwise it contains the opposite rule

(3) for every $\mathbf{X}\in\mathcal{X}$: (a) N_I contains $\bar{y}_1\ldots\bar{y}_p\!:\!x>\bar{x}$, (b) if $I(X)=\top$ then N_I contains $y_1\ldots y_p\!:\!x>\bar{x}$, otherwise it contains the opposite rule, and (c) for all other assignments u to $\mathsf{pa}(X)$, N_I contains $u\!:\!\bar{x}>x$.

We show that N_I entails $o>o'$ if and only if I satisfies ϕ. Clearly, we can reason on sets $\{X, Y_1, \ldots, Y_p, Z_{Y_1}, \ldots, Z_{Y_p}\}$ independently. So assume first that I satisfies $(X\vee Y_1)\wedge\cdots\wedge(X\vee Y_p)$.

If I satisfies X, then I entails $o\succ o'$ using the worsening flips $z_{Y_1}>\bar{z}_{Y_1}, \ldots, z_{Y_p}>\bar{z}_{Y_p}$ and $y_1\ldots y_p\!:\!x>\bar{x}$ (which can be performed in any order). Otherwise, I must satisfy $Y_1\wedge\cdots\wedge Y_p$, hence I entails $o\succ o'$ using the flips $z_{Y_1}\!:\!y_1>\bar{y}_1, \ldots, z_{Y_p}\!:\!y_p>\bar{y}_p$, then the flip $\bar{y}_1\ldots\bar{y}_p\!:\!x>\bar{x}$, then the flips $z_{Y_1}>\bar{z}_{Y_1}, \ldots, z_{Y_p}>\bar{z}_{Y_p}$, and finally the flips $\bar{z}_{Y_1}:\bar{y}_1>y_1,\ldots,\bar{z}_{Y_p}:\bar{y}_p>y_p$.

The converse is shown similarly, and finally we have that N_I entails $o>o'$ if and only if I satisfies ϕ. Now by construction, each N_I built in this manner has a probability $1/2^n$ according to $\mathcal{N}$. Hence the probability with which $\mathcal{N}$ entails $o\succ o'$ is $m/2^n$ if and only if ϕ has m models, which completes the reduction. □

5 Complexity of Optimisation

We now show that optimisation with tree-structured PCP-nets is both computationally easy and simple. The first result even holds for the much more general class of acyclic PCP-nets.

Proposition 4. *The probability for a given outcome o to be optimal for a given acyclic PCP-net $\mathcal{N}$ can be computed in linear time $O(n)$.*

Proof. In the spirit of the "forward sweeping" procedure of Boutilier *et al.* [2004], it can be easily shown that o is optimal for a deterministic CP-net $N\propto\mathcal{N}$ if and only if N contains (1) the rule $o[X]>\bar{o}[X]$ for all root nodes X, and (2) the rule $o[\mathsf{pa}(X)]:o[X]>\bar{o}[X]$ for all other nodes X. It follows that the probability sought for is the product of the probabilities of all these rules, which can clearly be computed in time $O(n)$. □

Proposition 5. *The outcome with the maximal probability of being optimal for a given, tree-structured PCP-net $\mathcal{N}$ can be computed in linear time $O(n)$.*

Proof. The algorithm is a simple dynamic programming algorithm, operating bottom-up in the tree. First, given a leaf node X with parent Y, the algorithm determines the optimal assignment to X given $Y=y$, by taking the highest probability between rules $y:x>\bar{x}$ and $y:\bar{x}>x$, and similarly for $Y=\bar{y}$.

Now in the general case, given a variable Y with parent Z and children $X_1,\ldots,X_k$, the algorithm first considers the value z for Z, and given this value searches for the most probable assignment to $Y,X_1,\ldots,X_k$ and their descendants. This can be done efficiently by comparing (1) $p_y\times p_{y1}\times\cdots\times p_{yk}$, where p_y is the probability of the rule $z:y>\bar{y}$ and p_{yi} $(i=1,\ldots,k)$ is the previously computed probability of the best assignment to X_i and its descendants given $Y=y$, and (2) $p_{\bar{y}}\times p_{\bar{y}1}\times\cdots\times p_{\bar{y}k}$. Then the algorithm computes in a similar manner the probability of the most probable assignment given $Z=\bar{z}$, and based on this decides on the value y or $\bar{y}$ for each of $z,\bar{z}$. Clearly, when all variables have been examined, the algorithm has computed the desired outcome. □

6 Conclusion

We proposed a "probabilistic" extension of conditional preference networks (CP-nets) for representing the preferences of a group of individuals over a set of combinatorial objects, or for representing ill-known preferences. We studied the probabilistic counterparts of the main reasoning tasks for CP-nets, namely dominance testing and optimisation, from the algorithmical and complexity viewpoints. We gave efficient algorithms for tree-structured probabilistic CP-nets, and as a by-product we obtained a linear-time algorithm for dominance testing in standard, tree-structured CP-nets.

As studied here, the expressiveness of our formalism is limited in two aspects. First, assuming a common, tree-like structure is unrealistic in some applicative settings. As future work, we plan to extend our results, in particular using a notion inspired from treewidth. The second limitation is due to the fact that the probability distribution on deterministic CP-nets which is represented by a probabilistic CP-net, is by definition an independent one (with rules as random variables). So as to allow PCP-net to model more realistic distributions, we plan to extend the representation by separating the probability distribution from the structure. An obvious choice is to use a Bayesian networks over the rules induced by the structure as random variables. Even with simple networks, this would allow, for instance, to represent fact such as: 3/4 of those individuals who prefer x to $\bar{x}$ given y also prefer z to $\bar{z}$ given t,u. While one could fear a jump in complexity, it is worth noticing that our main result for tree-structured CP-nets goes through, in the sense that with such representation, computing the probability of $o\succ o'$ would amount to estimate the probability of 2^{2k^2} deterministic CP-nets, that is, to call an oracle for inference only a small number of times. This leaves hope that the framework can be extended to richer representations while preserving the low complexity of certain tasks.

Acknowledgements

We thank Jérôme Lang and anonymous referees for their valuable comments on previous versions of this paper. The work presented here benefited of the support of the Agence Nationale de la Recherche, under grants ANR-11-BS02-008 and ANR-10-BLAN-0215.

References

[Boutilier *et al.*, 2001] Craig Boutilier, Fahiem Bacchus, and Ronen I. Brafman. Ucp-networks: A directed graphical representation of conditional utilities. In *UAI*, pages 56–64, 2001.

[Boutilier *et al.*, 2004] Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, Holger H. Hoos, and David Poole. Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *JAIR*, 21:135–191, 2004.

[Conitzer *et al.*, 2011] Vincent Conitzer, Jérôme Lang, and Lirong Xia. Hypercubewise preference aggregation in multi-issue domains. In *IJCAI*, pages 158–163, 2011.

[Cornelio, 2012] Cristina Cornelio. Dynamic and probabilistic cp-nets. Master's thesis, University of Padova, 2012.

[de Amo *et al.*, 2012] Sandra de Amo, Marcos L. P. Bueno, Guilherme Alves, and Nádia Félix F. da Silva. Cprefminer: An algorithm for mining user contextual preferences based on bayesian networks. In *ICTAI*, pages 114–121, 2012.

[Flum and Grohe, 2006] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory (Texts in Theoretical Computer Science. An EATCS Series)*. Springer-Verlag, 2006.

[Gelle and Weigel, 1996] Esther Gelle and Rainer Weigel. Interactive configuration using constraint satisfaction techniques. In *PACT*, pages 37–44, 1996.

[Goldsmith *et al.*, 2005] Judy Goldsmith, Jérôme Lang, Miroslaw Truszczyński, and Nic Wilson. The computational complexity of dominance and consistency in cp-nets. In *IJCAI*, pages 144–149, 2005.

[Gonzales *et al.*, 2008] Christophe Gonzales, Patrice Perny, and Sergio Queiroz. Preference aggregation with graphical utility models. In *AAAI*, pages 1037–1042, 2008.

[Koriche and Zanuttini, 2009] Frédéric Koriche and Bruno Zanuttini. Learning conditional preference networks with queries. In *IJCAI*, 2009.

[Li *et al.*, 2011] Minyi Li, Quoc Bao Vo, and Ryszard Kowalczyk. Majority-rule-based preference aggregation on multi-attribute domains with cp-nets. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS, 2011.

[Rao and Georgeff, 1991] Anand S. Rao and Michael P. Georgeff. Modeling rational agents within a bdi-architecture. In *KR*, 1991.

[Vadhan, 2002] Salil P. Vadhan. The complexity of counting in sparse, regular, and planar graphs. *SIAM J. Comput.*, 31(2):398–427, February 2002.

[Xia *et al.*, 2008] Lirong Xia, Vincent Conitzer, and Jérôme Lang. Voting on multiattribute domains with cyclic preferential dependencies. In *AAAI*, pages 202–207, 2008.

Boosting in the presence of label noise

Jakramate Bootkrajang
School of Computer Science
University of Birmingham
Birmingham, B15 2TT
United Kingdom

Ata Kabán
School of Computer Science
University of Birmingham
Birmingham, B15 2TT
United Kingdom

Abstract

Boosting is known to be sensitive to label noise. We studied two approaches to improve AdaBoost's robustness against labelling errors. One is to employ a label-noise robust classifier as a base learner, while the other is to modify the AdaBoost algorithm to be more robust. Empirical evaluation shows that a committee of robust classifiers, although converges faster than non label-noise aware AdaBoost, is still susceptible to label noise. However, pairing it with the new robust Boosting algorithm we propose here results in a more resilient algorithm under mislabelling.

1 Introduction

It is well known to practitioners that boosting is sensitive to label noise. The issue stems directly from the fundamental concept of boosting in that the effort is directed towards classifying the difficult samples. In fact, the complexity of the traditional boosting is very high, so much so that for a dataset with any configuration of its labels, it is possible to draw a decision boundary with zero training error. This seems to be a good approach to the classification problem if the difficult samples are not mislabelled samples in the first place. In reality however, there are no firm guarantees about the correctness of the class labels provided with the training set. In many applications, such as e.g. crowdsourcing data, and certain biomedical data, perfect training labels are almost impossible to obtain. A seemingly straightforward way to control boosting's complexity is by means of regularisation. However, regularisation alone might not be enough to solve this issue – as pointed out in Long & Servedio (2010), random misclassification defeats all boosters that optimise a convex objective. Yet, rather curiously, almost all of the existing *robust* boosters are still optimising a convex exponential loss. These boosters include the LogitBoost by Friedman et al. (1998) that optimises the binary log-loss, the Gentle-AdaBoost by Friedman et al. (1998) that is more stable because of a more conservative update step; the Modest-AdaBoost by Vezhnevets & Vezhnevets (2005) which penalises the ensemble when it makes a correct prediction on previously correctly predicted instances; the BB algorithm by Krieger et al. (2001) in which bagging is combined with boosting to average out the adverse effect of noisy labelled data. There is also a heuristic approach by Karmaker & Kwek (2006) where too difficult samples, i.e., those with very high weights, are removed from the training set according to a predefined threshold.

Motivated by the finding of Long and Servedio, Freund (2009) proposed a more robust boosting algorithm which optimises a non-convex potential function instead of the traditional exponential loss function. The general idea is to incorporate an early stopping as well as a mechanism to give up if the instance is to far away on the wrong side of the decision boundary. It shows promising results but unfortunately the boosting process becomes more complicated in that it also introduces a free parameter that has to be tuned. Freund suggests using cross-validation to tune the parameter however we can not rely on the cross-validation if our labels are noisy, unless we have a trusted validation set with correct labels.

Inspired by the work of Long and Servedio and Freund, we propose a different modification to AdaBoost for tackling label noise. We engineer our objective to be a combination of two complementary loss functions. Our new objective is somewhat related to those employed in the cost-sensitive boosting literature. However, in cost sensitive literature the cost associated with each instance is assumed to be given or known prior to the learning, and there is no label noise involved. By contrast, the primary task in robust boosting is to learn the mislabelling probabilities (which could be seen to be analogous to costs).

Recent developments on label-noise robust classifiers such as the robust Fisher Discriminant by Lawrence & Schölkopf (2001); Bouveyron & Girard (2009), the robust Logistic Regression by Bootkrajang & Kabán (2012); Raykar et al. (2010), the robust Gaussian Process by Hernández-Lobato et al. (2011) or the robust Nearest Neighbours by Barandela & Gasca (2000) suggest a new possibility to improve the existing booster without making any adjustment to the boosting algorithm by employing a robust classifier as a base learner. To the best of our knowledge, there are no attempts in the literature to pursue this direction and this is our starting point in this work.

To summarise, we investigate the solution to boosting in the presence of *random misclassification noise* at two different levels. At the lower level we study the robust committee where robust classifiers are combined and boosted using existing AdaBoost algorithm. At the higher level, we propose a new robust boosting algorithm that we call 'rBoost' where the objective function is a convex combination of two exponential losses. The coefficients of the combination represent uncertainty in the observed labels. The new boosting algorithm is closely related to AdaBoost and requires a relatively minor modification to the existing algorithm. Moreover our new objective is non-convex and exhibits robustness to labelling errors.

The paper is organised as follows. Section 2 reviews recent literature in label-noise robust classifiers and introduces the robust classifier that will be used throughout the paper. Section 3 presents the rBoost algorithm. Section 4 reports experimental results, and Section 6 draws conclusions of the study.

2 A robust base learner

In recent years many classifiers have been introduced to tackle the problem of learning in the presence of label noise. To date, there are a number of classifiers developed specifically for dealing with label noise: robust logistic regression, robust fisher discriminant, robust Gaussian Process or robust Nearest Neighbours. All of these can potentially be used as a base classifier, and it is then interesting to see how would such classifiers behave collectively in an ensemble. One way to construct a robust classifier is through a probabilistic latent variable model. Under the model, a robust classifier attempts to learn a posterior probability of the true labels via the likelihood of the observed labels.

We will deal with random label flipping noise, that is we assume the noise is independent of the specific features of individual data points, and flips the latent true label $y \in \{0,1\}$ from class k to class j into the observed label $\tilde{y} \in \{0,1\}$, with probability $\omega_{jk} := p(\tilde{y} = k|y = j)$. We define the likelihood of the observed label $\tilde{y}$ of a point $\mathbf{x}_i$ given the current parameter setting as the following:

$$\tilde{P}_i^k = p(\tilde{y} = k|\mathbf{x}_i, \theta, \{\omega_{jk}\}_{j,k=0}^1) = \sum_{j=0}^{1} \omega_{jk} p(y = j|\mathbf{x}_i, \theta) \tag{1}$$

that is, a linear combination of the 'true' class posteriors. From this assumption the modified log-likelihood is given by

$$\mathcal{L}(\theta, \{\omega_{jk}\}_{j,k=0}^1) = \sum_{i=1}^{n} \sum_{k=0}^{1} \mathbb{1}(\tilde{y}_i = k) \log(\tilde{P}_i^k) \tag{2}$$

where $\mathbb{1}(\cdot)$ is 1 if its argument is true and 0 otherwise, and n is the number of training points. Note that any probabilistic classifier yielding class posterior probability will fit the framework and can be converted into a robust classifier using the technique shown.

For the sake of concreteness we will employ logistic regression with parameter $\theta = \boldsymbol{\beta}$ in our study. We call the model 'robust Logistic Regression' (rLR), in which the likelihood of $\tilde{y} = 1$ is defined as:

$$\tilde{P}_i^1 = \omega_{11}\sigma(\boldsymbol{\beta}^T\mathbf{x}_i) + \omega_{01}(1 - \sigma(\boldsymbol{\beta}^T\mathbf{x}_i)) \tag{3}$$

Here, $\boldsymbol{\beta}$ is the weight vector orthogonal to the decision boundary and it determines the orientation of the separating plane and $\sigma(a) = 1/(1+\exp(-a))$ is the sigmoid function. Learning the robust logistic regression model involves estimating $\boldsymbol{\beta}$ and well as ω_{jk}. We follow the steps in Bootkrajang & Kabán (2012) where the conjugate gradient method is used to optimise $\boldsymbol{\beta}$. The gradient of the log-likelihood w.r.t the weight vector is, $\nabla_{\boldsymbol{\beta}}\mathcal{L}(\theta, \{\omega_{jk}\}_{j,k=0}^1) =$

$$\sum_{i=1}^{n} \Bigg[\left(\frac{\tilde{y}_i(\omega_{11} - \omega_{01})}{\tilde{P}_i^1} + \frac{(1 - \tilde{y}_i)(\omega_{10} - \omega_{00})}{\tilde{P}_i^0} \right) \times \sigma(\boldsymbol{\beta}^T\mathbf{x}_i)(1 - \sigma(\boldsymbol{\beta}^T\mathbf{x}_i)) \times \mathbf{x}_i \Bigg] \tag{4}$$

The following multiplicative updates are then used to estimate ω_{jk}:

$$\omega_{10} = \frac{g_{10}}{g_{10} + g_{11}}, \quad \omega_{11} = \frac{g_{11}}{g_{10} + g_{11}} \tag{5}$$

$$\omega_{00} = \frac{g_{00}}{g_{00} + g_{01}}, \quad \omega_{01} = \frac{g_{01}}{g_{00} + g_{01}} \tag{6}$$

where

$$g_{11} = \omega_{11} \sum_{i=1}^{n} \left(\frac{\tilde{y}_i \sigma(\boldsymbol{\beta}^T \mathbf{x}_i)}{\tilde{P}_i^1} \right)$$

$$g_{10} = \omega_{10} \sum_{i=1}^{n} \left(\frac{(1-\tilde{y}_i) \sigma(\boldsymbol{\beta}^T \mathbf{x}_i)}{\tilde{P}_i^0} \right)$$

$$g_{01} = \omega_{01} \sum_{i=1}^{n} \left(\frac{\tilde{y}_i (1-\sigma(\boldsymbol{\beta}^T \mathbf{x}_i))}{\tilde{P}_i^1} \right)$$

$$g_{00} = \omega_{00} \sum_{i=1}^{n} \left(\frac{(1-\tilde{y}_i)(1-\sigma(\boldsymbol{\beta}^T \mathbf{x}_i))}{\tilde{P}_i^0} \right) \tag{7}$$

3 The Robust Boosting

Suppose we have a training set with corrupted labels $D = \{(\mathbf{x}_i, \tilde{y}_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \Re^m$ and $\tilde{y}_i \in \{+1, -1\}$. Let a base hypothesis be a decision function $h : \mathbf{x} \rightarrow \tilde{y}$. Under the boosting framework, a final hypothesis is a linear combination of the base hypotheses and it takes the following additive form:

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}) \tag{8}$$

In boosting, the 0/1 misclassification loss incurred by the final hypothesis is measured by the exponential loss:

$$\sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1) e^{-H(\mathbf{x}_i)} + \mathbb{1}(\tilde{y}_i = -1) e^{H(\mathbf{x}_i)} \tag{9}$$

This forms a boosting objective that has to be optimised. However, in the situation where labels are contaminated the loss in eq.(9) is not ideal, for obvious reasons. Instead, we form a new objective that explicitly takes into account uncertainties in labels:

$$\sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1) \Big\{ \gamma_{00} e^{-H(\mathbf{x}_i)} + \gamma_{01} e^{H(\mathbf{x}_i)} \Big\} + \mathbb{1}(\tilde{y}_i = -1) \Big\{ \gamma_{11} e^{H(\mathbf{x}_i)} + \gamma_{10} e^{-H(\mathbf{x}_i)} \Big\} \tag{10}$$

Here, $\gamma_{jk} = p(\tilde{y} = k | y = j)$ are probabilistic factors representing uncertainties in labels. Intuitively, the loss is weighed up or down depending on the gamma parameters γ_{jk}. For example, $\gamma_{01} = 0.3$ and $\gamma_{10} = 0$ indicates the situation where labels in the negative class (or class 0) are all correct – because no flipping from positive to negative occurred – but labels in the positive class are contaminated. Accordingly, the new loss accounts for this by adjusting the loss for the positive class (class 1) to: $0.7 * e^{-H} + 0.3 * e^{H}$. This is a hyperbolic cosine with the two tails adjusted and it represents the modified loss associated with the positive class. The shapes of such modified loss functions are depicted in Figure 1. From the figure we see that the classification that is 'too correct' will be penalised, hence reducing the overfitting problem. Meanwhile the loss of the negative class (class 0), which is $e^H + 0 * e^{-H} = e^H$, reduces to traditional boosting. It may be interesting to note that a similar shape of the loss can also be obtained by truncating the Taylor expansion of the exponential function to some finite degree. This could also be used to implement the same idea, although it would not have the transparent formulation given above.

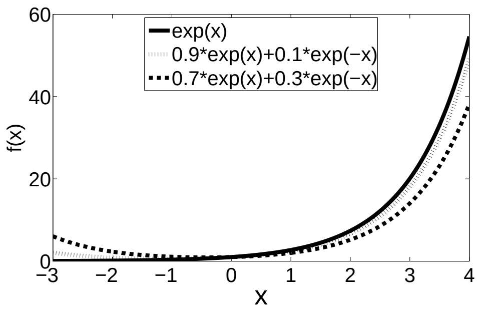


Figure 1: Various setups of the Γ and their associated loss shape.

3.1 Adding a new base learner $h_t(\cdot)$

Consider the case $\tilde{y} = 1$, and define $d_{00} = e^{-H(\mathbf{x})}$, $d_{01} = e^{H(\mathbf{x})}$. Likewise, when $\tilde{y} = -1$ define $d_{11} = e^{H(\mathbf{x})}$, $d_{10} = e^{-H(\mathbf{x})}$ to be an unnormalised distribution of the data $(\mathbf{x}, \tilde{y})$. It can be shown that at the iteration t of boosting, minimising the loss in eq.(10) w.r.t the new $h_t(\mathbf{x})$ is equivalent to minimising the following (the derivation details are given in the Appendix):

$$\arg\min_{h,\alpha} \; 2\sinh(\alpha) \sum_{i=1}^{n} \Big\{ w_i \mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i) \Big\} + e^{-\alpha} \sum_{i=1}^{n} \Big\{ \mathbb{1}(\tilde{y}_i = 1) w_{00} + \mathbb{1}(\tilde{y}_i = -1) w_{11} \Big\} + e^{\alpha} \sum_{i=1}^{n} \Big\{ \mathbb{1}(\tilde{y}_i = 1) w_{01} + \mathbb{1}(\tilde{y}_i = -1) w_{10} \Big\} \tag{11}$$

where

$$w_i = \begin{cases} (w_{00} - w_{01}), & \text{if } \tilde{y}_i = +1. \\ (w_{11} - w_{10}), & \text{if } \tilde{y}_i = -1. \end{cases} \tag{12}$$

and

$$w_{jk} = \gamma_{jk} \cdot d_{jk} \tag{13}$$

From this, it is immediate to see that in order to minimise the loss we have to seek for $h_t(\mathbf{x})$ that minimises the misclassification error $\epsilon_t = \sum_{i=1}^{n} w_i \mathbb{1}(\tilde{y}_i \neq h(\mathbf{x}_i))$.

This step is identical to the traditional AdaBoost except that the misclassification error of the current classifier is measured against different weighting factors which take into account the uncertainty of the observed noisy label as indicated by γ_{jk}. Note that the expression is fully compatible with the traditional AdaBoost such that the rBoost reduces to the original AdaBoost when $\gamma_{01} = 0$ and $\gamma_{10} = 0$. We emphasise that the weights in the rBoost need not be normalised. In fact in the original AdaBoost the normalisation simply facilitates the algebra in deriving a closed-form update for α_t.

3.2 Updating α_t

Now in our case, to get the update for α_t we take derivative of eq.(11) w.r.t α_t, equate it to zero:

$$\begin{aligned} 2\cosh(\alpha)\sum_{i=1}^{n}\Big\{w_i \mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\Big\} \\ - e^{-\alpha}\sum_{i=1}^{n}\Big\{\mathbb{1}(\tilde{y}_i = 1)w_{00} + \mathbb{1}(\tilde{y}_i = -1)w_{11}\Big\} \\ + e^{\alpha}\sum_{i=1}^{n}\Big\{\mathbb{1}(\tilde{y}_i = 1)w_{01} + \mathbb{1}(\tilde{y}_i = -1)w_{10}\Big\} = 0 \end{aligned} \tag{14}$$

Now, this equation cannot be solved in closed form. We resort to numerical optimisation to solve for the α_t. Note the term which gets multiplied by $2\cosh(\alpha)$ is nothing but our error ϵ_t defined earlier.

3.3 Updating the sample weights

Next, to derive the update for the weight vectors, recall that we define $w_{jk} = \gamma_{jk}e^{-\tilde{y}_i H(\mathbf{x}_i)}$. It follows, for example, that the update for w_{00} can be written as:

$$\begin{aligned} w_{00}^{t+1} &= \gamma_{00}e^{-\tilde{y}_i(H+\alpha h)} \\ &= \gamma_{00}e^{-\tilde{y}_i H} \cdot e^{-\tilde{y}_i \alpha h} \\ &= \gamma_{00}d_{00}^{t} \cdot e^{\alpha(2\mathbb{1}(h(\mathbf{x}_i)\neq\tilde{y}_i)-1)} \\ &= \gamma_{00}d_{00}^{t} \cdot e^{2\alpha\mathbb{1}(h(\mathbf{x}_i)\neq 1)} \cdot e^{-\alpha} \\ &\propto \gamma_{00}d_{00}^{t} \cdot e^{2\alpha\mathbb{1}(h(\mathbf{x}_i)\neq 1)} \end{aligned} \tag{15}$$

where we used the rewriting: $-\tilde{y}h = 2\mathbb{1}(h(\mathbf{x}) \neq \tilde{y}) - 1$; and since $e^{-\alpha}$ are shared among all w_{jk} it does not affect the optimisation. Similarly for the rest of the weight vectors we get:

$$w_{01}^{t+1} = \gamma_{01}d_{01}^{t} \cdot e^{2\alpha\mathbb{1}(h(\mathbf{x}_i)\neq -1)} \tag{16}$$

$$w_{11}^{t+1} = \gamma_{11}d_{11}^{t} \cdot e^{2\alpha\mathbb{1}(h(\mathbf{x}_i)\neq -1)} \tag{17}$$

$$w_{01}^{t+1} = \gamma_{10}d_{10}^{t} \cdot e^{2\alpha\mathbb{1}(h(\mathbf{x}_i)\neq 1)} \tag{18}$$

One way to implement this is to keep the distribution d_{jk} separately and multiply it by γ_{jk} to get a new w_{jk} in each iteration.

3.4 Updating γ_{jk}

Finally, as mentioned earlier, we would also like to estimate the label flipping coefficients, γ_{jk}. We could take derivative of the loss incurred by the current ensemble, eq.(10), w.r.t each gamma and try to solve this direcly. This did not yield satisfactory results in our experience, most likely because the loss lacks probabilistic semantics. The workaround is to convert the output of boosting i.e. H into a probability. There are three popular approaches to do that: 1) Logistic calibration $p(y = 1|x, H) = 1/(1 + \exp(-H))$ Friedman et al. (1998), 2) Platt's calibration $p(y = 1|x, H) = 1/(1 + \exp(AH + B))$ where A and B need to be learnt Platt (1999), and 3) Isotronic regression Robertson (1988). Niculescu-Mizil & Caruana (2005) empirically shows that Platt's technique and Isotronic regression are superior to a simple logistic transform. In addition, Platt's method has a slight advantage over IsoReg on small sample size. Hence, in this study, we will employ Platt's method to get calibrated posterior probabilities.

By converting H to $p(y = 1|x, H)$, we can estimate the gamma from the following binomial log-loss, or cross-entropy. Using the notation $P(x) = p(y = 1|x, H)$ and $\bar{P}(x) = 1 - P(x)$, this is:

$$\begin{aligned} -\sum_{i=1}^{n}\mathbb{1}(\tilde{y}_i = 1)\log\Big\{\gamma_{11}P(\mathbf{x}_i) + \gamma_{01}\bar{P}(\mathbf{x}_i)\Big\} \\ + \mathbb{1}(\tilde{y}_i = -1)\log\Big\{\gamma_{00}\bar{P}(\mathbf{x}_i) + \gamma_{10}P(\mathbf{x}_i)\Big\} \end{aligned} \tag{19}$$

Following the Lagrangian method which imposes $\gamma_{00} + \gamma_{01} = 1$ and $\gamma_{11} + \gamma_{10} = 1$, similarly to the technique in the latent variable model (outlined in Section 2), the multiplicative updates for γ_{jk} are found to be:

$$\gamma_{10} = \frac{g_{10}}{g_{10} + g_{11}}, \quad \gamma_{11} = \frac{g_{11}}{g_{10} + g_{11}} \tag{20}$$

$$\gamma_{00} = \frac{g_{00}}{g_{00} + g_{01}}, \quad \gamma_{01} = \frac{g_{01}}{g_{00} + g_{01}} \tag{21}$$

where

$$\begin{aligned} g_{11} &= \gamma_{11}\sum_{i=1}^{n}\left(\frac{\mathbb{1}(\tilde{y}_i = 1)P_i}{\gamma_{11}P_i + \gamma_{01}\bar{P}_i}\right) \\ g_{10} &= \gamma_{10}\sum_{i=1}^{n}\left(\frac{\mathbb{1}(\tilde{y}_i = -1)P_i}{\gamma_{10}P_i + \gamma_{00}\bar{P}_i}\right) \\ g_{01} &= \gamma_{01}\sum_{i=1}^{n}\left(\frac{\mathbb{1}(\tilde{y}_i = 1)\bar{P}_i}{\gamma_{11}P_i + \gamma_{01}\bar{P}_i}\right) \\ g_{00} &= \gamma_{00}\sum_{i=1}^{n}\left(\frac{\mathbb{1}(\tilde{y}_i = -1)\bar{P}_i}{\gamma_{10}P_i + \gamma_{00}\bar{P}_i}\right) \end{aligned} \tag{22}$$

Our method is summarised in Algorithm 1.

As mentioned in the introduction, our rBoost algorithm has some analogies with cost-sensitive boosting Fan & Stolfo (1999); Masnadi-Shirazi & Vasconcelos (2011). One major difference is that the weighting factors in our case are outside of the exponential, whereas they are inside the exp in the mentioned works. In Fan & Stolfo (1999) the author did briefly discuss the possibility of having the weighting factors outside of the exponential, however their update of the weight is different from ours. Besides, the goal of cost-sensitive methods is different from ours. In cost-sensitive framework the cost is assumed to be known or given by the expert, and there is no implication of labelling errors.

Algorithm 1 rBoost

```
**Input:** data $\{\mathbf{x}, \tilde{y}\}^n$, boosting round $T$
Initialize $w_{jk} = \gamma_{jk}$
**for** $t = 1$ **to** $T$ **do**
    (1)$h_t = \arg\max_{\boldsymbol{\beta}}$ eq.(2) weighted by $w_i$.
    (2)Calculate the error w.r.t $w_i$ defined in eq.(12)
        $\epsilon_t = \sum_{i=1}^{n} w_i \mathbb{1}(\tilde{y}_i \neq h_t(\mathbf{x}_i))$
    (3)Optimise $\alpha_t$ numerically using the gradient in eq.(14)
    (4)Update $w_{jk}$ according to eq.(15)–(18).
    (5)Calculate $p(y = 1|\mathbf{x}, H)$ using Platt's method.
    (6)Update $\gamma_{jk}$ using eq.(20)-(21).
**end for**
Output the final classifier $sign(\sum_{t=1}^{T} \alpha_t h_t)$.
```

4 Empirical Evaluation

This section will investigate the performance of our robust boosting methods in practice. In addition, our new rBoost algorithm will be compared to the standard AdaBoost, GentleAdaBoost and ModestAdaBoost.

4.1 Methodology

We will study 4 configurations of base-learner and booster pairs: 1) LR + AdaBoost, 2) rLR + AdaBoost, 3) LR + rBoost and 4) rLR + rBoost. These four combinations will shed light on whether 1) a robust committee is robust against label errors?, 2) the new rBoost can counteract the bad effects of label noise? and finally 3) What can we get from pairing them together? We set our baseline to be the GentleAdaBoost and ModestAdaBoost where the base learner is a decision tree with maximum node splits of 2. For LR to serve as a weak learner, we employ random subsampling to create diversity in the ensemble. Further, we create two types of training sets by artificially injecting symmetric and asymmetric label noise at rate 10%, as well as at rate 30% into the training data. We train on the corrupted training set and validate the performance of the ensemble on a clean test set. We report the average and standard deviation of the misclassification rates from 10 independent random repetitions of 150 rounds of boosting each.

Table 1: Characteristic of the dataset used.

Data set	# of pos. samp.	# of neg. samp.	dim.
Banana	2375(45%)	2924(55%)	2
Diabetes	268(35%)	500(65%)	8
Heart	120(44%)	150(56%)	13
Image	1188(57%)	898(43%)	18
Titanic	14(58%)	10(42%)	3
Twonorm	3703(50%)	3697(50%)	20
Waveform	1647(33%)	3353(67%)	21

4.2 Datasets

We select seven UCI machine learning datasets Frank & Asuncion (2010) namely Banana, Diabetes, Heart, Image, Twonorm and Waveform to use to evaluate the proposed boosting combinations. We use 80% of the dataset for training and 20% for testing purpose. The characteristics of the datasets used are summarised in Table 1.

4.3 Results

We first investigate the behaviour of the robust classifiers as weak learners within the original AdaBoost algorithm. From the leftmost column of Tables 2-5, we see that when a robust classifier is used as a base learner the generalisation error of the ensemble is already lower in comparison to the original non-robust AdaBoost in 4 out of 7 datasets. The finding is consistent across all noise levels. It is very interesting to observe this because even though the base classifier is robust, it is still under the control of the original AdaBoost. Namely, the boosting will still guide the classifiers to focus on the more difficult parts of the dataset (which of course are likely to contain the points whose labels are wrong). Why is then this committee of robust classifiers more accurate? Lower error can come from two different sources: Either the robust committee is indeed robust against labelling errors, or it simply converges faster. To check this we run both configurations for more rounds to see the dynamics of the ensemble. Plotted in Figure 2 are the training and test errors of AdaBoost using the robust classifiers (rLR) as well as using the traditional classifiers (LR) on selected datasets. Superimposed for reference are ModestAdaBoost and rLR + rBoost.

It turns out that the robust committee converges much quicker than the non-robust committee. However when boosted long enough we are starting to see that their classification performances become very similar. This answers our first research question. The robust classifiers as weak learners introduce what is understood to be a 'good diversity' in the ensemble, and drives the ensemble to convergence much quicker than the non-robust committee. Unfortunately however, the robustness of the base learner is not enough to withstand the effect of labelling errors.

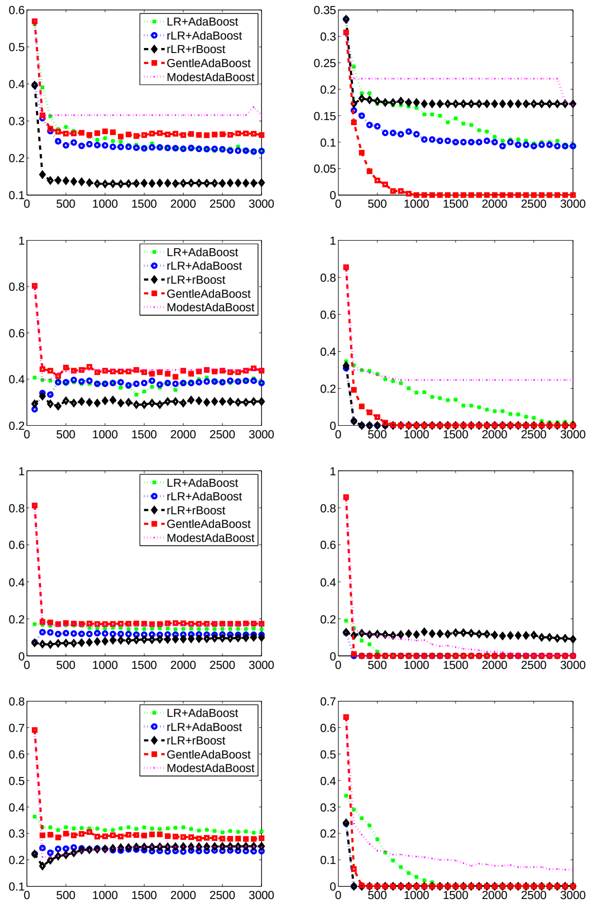


Figure 2: Test error(left) and traing error(right) for 'Banana','Diabetes','Twonorm','Waveform' datasets. The x-axis indicates boosting rounds while the y-axis shows classification errors.

Now we see that having rLR as a base learner alone is not enough to counteract the bad effect of mislabelling. We investigate further if we can pair rLR which has a fast convergence rate with our new rBoost algorithm.

Before proceeding, we need to establish that rBoost is superior to original AdaBoost when there is label noise. To this end, we consider two combinations: 1) AdaBoost+LR and 2)rBoost + LR in Tables 2-5. From the tables we see that rBoost+LR performs comparably to its non-robust booster counterpart when the noise rate is relatively low, and in the case of symmetric label case (i.e. the easy cases in terms of label noise). However when the noise is asymmetric and more severe (Table 5), rBoost significantly outperforms the original AdaBoost in all the cases tested. This answers our second research question. That is, rBoost significantly improves over the original AdaBoost in terms of classification performance especially in higher label contamination rate conditions and in asymmetric label noise conditions (i.e. the difficult cases).

Next, we equip our rBoost method with the robust base classifiers that enjoy fast convergence to obtain our final robust boosting algorithm. These results are shown in the fourth column of Table 5. The superior performance of this approach is most apparent, and we also give an illustrative example of the working of our rBoost on the 'Banana' dataset in Figure 3. We see that the original AdaBoost generated a patchy decision boundary as a result of label noise, while our rBoost returned a smoother and more appropriate decision boundary.

Further, we validate our approach for estimating the flip probabilities γ_{jk} using the multiplicative updates given in eq.(20) and eq.(21). Disappointingly, we see that the results (5th and 6th column of Tables 2-5) are not as good as the ideal setting where the γ_{jk} are fixed to the true value (rBoost-Fixed gamma). However, and more interestingly, we observe that the quality of the estimated gammas depends highly on the quality of the calibrated probability used in the update. Assuming that we have a trusted validation set that we can use to obtain a more accurate calibrated probability, we ask how well can we estimate the gammas? We hold out a small subset of the dataset, where all of the labels are clean. This will be our trusted validation set, and we took this set as tiny as 20 points only. We feed this small trusted dataset into the Platt's calibration method. We carried out this experiment on Banana, Image and Twonorm. The classification error from 10 repeated runs of our rBoost algorithm with the use of the trusted validation set as a source for calibrating the probability is 15.74±0.23% on 'Banana' at 30% asymmetric noise, compared to 23.83% without. This is taken from the sixth column of Table 5. On Image at 30% asymmetric noise the error is as low as 7.61±0.19% and on Twonorm it is 9.73±0.31%. Intriguingly, a tiny trusted set of 20 points is able to improve the situation even for the Image data, where the training set size is as large as 1300 (80% of total number of samples in Image). Thus we can conclude that the trusted validation set approach may be seen as a technique to effectively and efficiently incorporate extra knowledge about the labels into the rBoost algorithm. We should note, this differs from simply including the trusted samples into the training set, since the latter would simply make a slight reduction of the noise rate. Of course, the larger the trusted validation set for calibration, the better probability calibration we can expect, and consequently this should lead to more accurate estimates of the gammas (γ_{jk}), and hence to better classification performance.

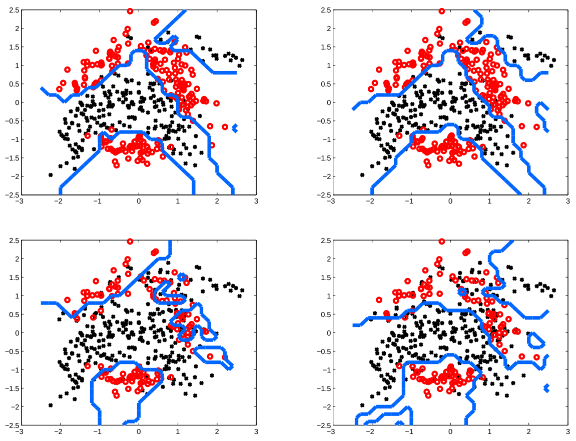


Figure 3: Comparison of the decision boundaries obtained from AdaBoost(left) and rBoost(right) in noise-free case(top) and 30% asymmetric noise case(bottom) on banana dataset.

5 Discussion

We have assumed throughout the study that the label noise is random and it occurs independently from the input sample. Worth mentioning that there exist other types of noise such as a non-random label noise, malicious or adverserial noise, which may require a different treatment. The random noise treated here is simpler and more generic as it does not require special knowledge about the noise process. By contrast, modelling a non-random noise requires us to encode domain expertise into the formulation, and as a consequence it will yield a model specific to the application. Interestingly, despite its simplicity, the random noise model finds its use even in the cases where the random assumption does not perfectly hold true, as it microarray anaylsis (Bootkrajang & Kabán (2013)). In the context of boosting, there is an attempt to tackle non-random noise in Takenouchi et al. (2008) where both binary and multi-class problems are investigated.

6 Conclusion

We presented a robust boosting algorithm based on the famous AdaBoost algorithm, which we called rBoost. The rBoost has some advantageous properties, namely its objective is non-convex, hence more robust, and it incorporates label noise parameters that can be estimated efficiently using the proposed multiplicative update rules. The new algorithm is also appealing since it requires a minor modification to the existing AdaBoost algorithm. We further demonstrated that the label noise parameters can be more accurately estimated by using a trusted validation set for Platt's calibration algorithm as a form of extra information. It shows good result close to the rBoost with label noise parameters fixed to the true values. In addition, we have empirically shown that simply employing a robust classifier as a base learner in the AdaBoost does not help alleviating the bad effect of label noise. However, rather interestingly, its effect is to speed up the boosting process. This could be advantageous in cases of low noise. An intriguing future direction are the theoretical analysis of our proposed rBoost and extensions to multi-class problems.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments and suggestions, and the School of Computer Science, University of Birmingham for providing computing facilities for this study. JB thanks the Royal Thai Government for financial support.

Appendix

This section shows derivation details of the rBoost algorithm. The loss of the rBoost is defined as:

$$\mathcal{L}(H) = \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\Big\{\gamma_{00}e^{-H(\mathbf{x}_i)} + \gamma_{01}e^{H(\mathbf{x}_i)}\Big\} + \mathbb{1}(\tilde{y}_i = -1)\Big\{\gamma_{11}e^{H(\mathbf{x}_i)} + \gamma_{10}e^{-H(\mathbf{x}_i)}\Big\} \quad (23)$$

Here, γ_{jk} are again probabilistic factors representing uncertainty in labels. We write out the form of $H(\mathbf{x}_i)$ for the next round of AdaBoost. Minimising this loss of eq. (23) in a step-wise manner is then equivalent to minimising the following:

$$\arg\min_{h,\alpha} \Big(\sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\Big\{\gamma_{00}e^{-(H(\mathbf{x}_i)+\alpha h(\mathbf{x}_i))} + \gamma_{01}e^{H(\mathbf{x}_i)+\alpha h(\mathbf{x}_i)}\Big\} + \mathbb{1}(\tilde{y}_i = -1)\Big\{\gamma_{11}e^{H(\mathbf{x}_i)+\alpha h(\mathbf{x}_i)} + \gamma_{10}e^{-(H(\mathbf{x}_i)+\alpha h(\mathbf{x}_i))}\Big\}\Big) \quad (24)$$

$$= \arg\min_{h,\alpha} \sum_{i=1}^{n} \Big\{\mathbb{1}(\tilde{y}_i = 1)\gamma_{00}e^{-H(\mathbf{x}_i)}e^{-\alpha h(\mathbf{x}_i)} \quad (25)$$

$$+ \mathbb{1}(\tilde{y}_i = 1)\gamma_{01}e^{H(\mathbf{x}_i)}e^{\alpha h(\mathbf{x}_i)} \quad (26)$$

$$+ \mathbb{1}(\tilde{y}_i = -1)\gamma_{11}e^{H(\mathbf{x}_i)}e^{\alpha h(\mathbf{x}_i)} \quad (27)$$

$$+ \mathbb{1}(\tilde{y}_i = -1)\gamma_{10}e^{-H(\mathbf{x}_i)}e^{-\alpha h(\mathbf{x}_i)}\Big\} \quad (28)$$

Table 2: Average classification errors and their standard deviations for AdaBoost and rBoost at 10% symmetric noise. Boldface font shows the result which is statistically significant as tested with Wilcoxon ranksum test at the 5% level.

Dataset	AdaBoost		rBoost-Fixed gamma		rBoost		Gentle	Modest
	LR	rLR	LR	rLR	LR	rLR	Boost	Boost
Banana	18.53±1.0	13.13±1.1	17.53±1.8	12.94±0.9	17.44±1.8	12.96±0.9	16.09±1.6	21.87±3.4
Diabetes	24.00±2.7	25.80±2.3	24.10±1.7	25.63±1.5	23.87±1.9	25.20±2.4	27.40±2.0	24.33±1.9
Heart	21.20±4.4	21.60±3.1	22.40±3.9	20.30±3.5	22.10±4.8	20.90±4.4	23.60±3.1	22.40±3.5
Image	14.61±1.3	4.08±1.0	15.29±1.5	4.49±0.8	13.51±1.4	4.12±0.8	4.43±0.7	15.91±3.1
Titanic	22.76±1.3	22.32±1.1	22.97±1.4	22.37±1.1	22.76±1.2	22.37±1.4	22.30±1.7	23.28±1.4
Twonorm	5.78±0.8	4.30±0.8	5.75±0.7	4.42±0.9	5.72±1.0	4.41±0.7	9.65±1.0	7.21±0.5
Waveform	16.67±1.5	**13.40±0.7**	16.43±0.7	14.65±0.8	16.12±1.2	**13.47±0.6**	14.98±0.8	14.77±1.5
All	17.65±1.8	**14.95±1.5**	17.78±1.6	**14.97±1.3**	17.36±1.9	**14.78±1.6**	16.92±1.5	18.54±2.2

Table 3: Average classification errors and their standard deviations for AdaBoost and rBoost at 30% symmetric noise.

Dataset	AdaBoost		rBoost-Fixed gamma		rBoost		Gentle	Modest
	LR	rLR	LR	rLR	LR	rLR	Boost	Boost
Banana	18.71±3.1	14.73±3.0	18.14±1.7	14.47±2.1	18.05±1.6	14.94±2.7	20.62±1.6	24.69±2.5
Diabetes	25.37±2.3	27.47±1.9	24.90±2.3	29.57±2.4	25.23±2.7	28.57±2.3	30.60±2.9	26.67±2.3
Heart	22.10±5.7	21.50±4.0	22.70±4.0	22.60±6.5	22.90±4.9	21.90±4.3	30.00±5.5	24.80±3.7
Image	14.67±1.4	6.94±1.0	15.23±1.0	6.67±1.0	14.30±0.9	6.52±0.9	7.51±1.0	20.10±4.4
Titanic	23.12±1.6	23.01±1.8	23.27±1.5	22.92±1.4	23.09±1.4	23.11±1.8	22.80±1.9	23.41±1.3
Twonorm	8.53±1.1	6.67±0.9	8.77±1.0	6.87±1.3	8.63±1.0	6.60±1.1	16.06±2.0	8.84±0.9
Waveform	21.02±2.1	16.88±1.8	20.80±2.4	18.16±2.0	20.41±2.2	16.96±1.6	20.26±2.2	15.57±0.9
All	19.07±2.5	**16.74±2.1**	19.11±1.9	**17.32±2.4**	18.94±2.1	**16.94±2.1**	21.12±2.4	20.58±2.3

Now consider each term in the sum.

$$\begin{aligned}
(25) = &\sum_{i|h(\mathbf{x}_i)=\tilde{y}_i} \mathbb{1}(\tilde{y}_i = 1)\gamma_{00} e^{-H(\mathbf{x}_i)} e^{-\alpha} \\
&+ \sum_{i|h(\mathbf{x}_i)\neq\tilde{y}_i} \mathbb{1}(\tilde{y}_i = 1)\gamma_{00} e^{-H(\mathbf{x}_i)} e^{\alpha} \\
= &\sum_{i=1}^{n} \Big(1 - \mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\Big)\mathbb{1}(\tilde{y}_i = 1)\gamma_{00} e^{-H(\mathbf{x}_i)} e^{-\alpha} \\
&+ \sum_{i|h(\mathbf{x}_i)\neq\tilde{y}_i} \mathbb{1}(\tilde{y}_i = 1)\gamma_{00} e^{-H(\mathbf{x}_i)} e^{\alpha} \\
= &\sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\gamma_{00} e^{-H(\mathbf{x}_i)} e^{-\alpha} \\
&- \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\gamma_{00} e^{-H(\mathbf{x}_i)} e^{-\alpha} \\
&+ \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\gamma_{00} e^{-H(\mathbf{x}_i)} e^{\alpha} \\
= &\Big(e^{\alpha} - e^{-\alpha}\Big) \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\gamma_{00} e^{-H(\mathbf{x}_i)} \\
&+ \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\gamma_{00} e^{-H(\mathbf{x}_i)} e^{-\alpha} \qquad (29)
\end{aligned}$$

Using similar substitution and grouping, we also have the following:

$$\begin{aligned}
(26) = &\Big(e^{-\alpha} - e^{\alpha}\Big) \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\gamma_{01} e^{H(\mathbf{x}_i)} \\
&+ \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = 1)\gamma_{01} e^{H(\mathbf{x}_i)} e^{\alpha} \qquad (30)
\end{aligned}$$

$$\begin{aligned}
(27) = &\Big(e^{\alpha} - e^{-\alpha}\Big) \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = -1)\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\gamma_{11} e^{H(\mathbf{x}_i)} \\
&+ \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = -1)\gamma_{11} e^{H(\mathbf{x}_i)} e^{-\alpha} \qquad (31)
\end{aligned}$$

$$\begin{aligned}
(28) = &\Big(e^{-\alpha} - e^{\alpha}\Big) \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = -1)\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\gamma_{10} e^{-H(\mathbf{x}_i)} \\
&+ \sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = -1)\gamma_{10} e^{-H(\mathbf{x}_i)} e^{\alpha} \qquad (32)
\end{aligned}$$

Summing all four expressions we have the objective:

$$\begin{aligned}
&\arg\min_{h,\alpha}(e^{\alpha} - e^{-\alpha}) \sum_{i=1}^{n} \Big\{\mathbb{1}(\tilde{y}_i = 1)\gamma_{00} e^{-H(\mathbf{x}_i)}\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i) \\
&\quad + \mathbb{1}(\tilde{y}_i = -1)\gamma_{11} e^{H(\mathbf{x}_i)}\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\Big\} \\
&+ e^{-\alpha} \sum_{i=1}^{n} \Big\{\mathbb{1}(\tilde{y}_i = 1)\gamma_{00} e^{-H(\mathbf{x}_i)} + \mathbb{1}(\tilde{y}_i = -1)\gamma_{11} e^{H(\mathbf{x}_i)}\Big\} \\
&- (e^{\alpha} - e^{-\alpha}) \sum_{i=1}^{n} \Big\{\mathbb{1}(\tilde{y}_i = 1)\gamma_{01} e^{H(\mathbf{x}_i)}\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i) \\
&\quad + \mathbb{1}(\tilde{y}_i = -1)\gamma_{10} e^{-H(\mathbf{x}_i)}\mathbb{1}(h(\mathbf{x}_i) \neq \tilde{y}_i)\Big\} \\
&+ e^{\alpha} \sum_{i=1}^{n} \Big\{\mathbb{1}(\tilde{y}_i = 1)\gamma_{01} e^{H(\mathbf{x}_i)} + \mathbb{1}(\tilde{y}_i = -1)\gamma_{10} e^{-H(\mathbf{x}_i)}\Big\} \qquad (33)
\end{aligned}$$

Table 4: Average classification errors and their standard deviations for AdaBoost and rBoost at 10% asymmetric noise.

Dataset	AdaBoost		rBoost-Fixed gamma		rBoost		Gentle	Modest
	LR	rLR	LR	rLR	LR	rLR	Boost	Boost
Banana	17.85±2.7	13.54±1.3	16.78±1.7	12.55±1.1	17.81±2.4	13.65±1.2	16.81±1.5	22.27±2.8
Diabetes	24.70±1.6	25.67±1.6	24.27±1.7	25.57±1.5	24.23±1.6	25.97±1.7	27.80±2.1	24.93±1.6
Heart	21.70±4.4	21.50±3.9	21.60±3.7	22.20±2.9	21.30±3.7	21.10±2.7	26.10±3.2	21.70±4.2
Image	15.88±1.0	4.52±1.0	15.20±1.6	4.06±0.9	15.15±1.5	4.40±1.2	4.45±0.7	24.12±1.9
Titanic	22.87±1.3	23.06±1.3	22.65±1.3	22.23±1.1	23.58±1.6	22.44±1.0	22.83±1.5	23.63±1.5
Twonorm	7.02±1.6	5.21±1.1	6.16±1.4	4.51±0.8	6.56±1.4	5.34±1.1	9.19±1.0	7.71±1.2
Waveform	18.10±1.3	14.48±1.1	17.83±1.2	16.23±1.5	17.71±1.2	14.54±1.4	16.52±1.0	14.19±0.7
All	18.30±1.9	**15.42±1.6**	17.78±1.7	**15.33±1.4**	18.04±1.9	**15.34±1.5**	17.67±1.6	19.79±1.9

Table 5: Average classification errors and their standard deviations for AdaBoost and rBoost at 30% asymmetric noise.

Dataset	AdaBoost		rBoost-Fixed gamma		rBoost		Gentle	Modest
	LR	rLR	LR	rLR	LR	rLR	Boost	Boost
Banana	31.45±5.2	23.53±4.7	27.31±4.5	**14.27±1.0**	32.39±3.9	23.83±4.1	25.38±2.7	33.04±6.8
Diabetes	32.20±2.1	33.43±3.9	29.47±3.0	**30.20±2.6**	32.80±3.3	33.27±3.1	38.37±3.6	32.07±3.5
Heart	27.60±5.5	27.30±6.8	23.00±4.3	**24.30±3.8**	28.20±6.3	28.00±6.9	32.00±7.2	29.60±11.7
Image	22.48±1.6	10.70±0.9	16.96±1.8	**5.47 ±1.0**	20.53±1.6	9.82 ±1.5	11.94±1.1	26.44±1.3
Titanic	32.60±8.4	31.21±8.7	23.88±1.8	**22.14±1.5**	33.17±9.3	30.73±8.7	32.94±9.0	33.49±13.7
Twonorm	16.02±2.4	12.07±2.0	8.89 ±1.5	**6.51 ±1.3**	14.68±2.3	12.19±2.0	17.85±1.8	16.62±3.1
Waveform	28.83±2.8	23.43±2.5	24.27±2.1	**19.95±1.6**	28.39±3.1	23.02±2.5	27.31±2.5	21.10±2.2
All	27.31±3.9	23.09±4.2	21.96±2.7	**17.54±1.8**	27.16±4.2	22.97±4.1	26.54±3.9	27.47±6.0

$$\begin{aligned}
&= \arg\min_{h,\alpha} 2\sinh(\alpha)\times \\
&\sum_{i=1}^{n}\Big\{\mathbb{1}(\tilde{y}_i=1)\mathbb{1}(h(\mathbf{x}_i)\neq\tilde{y}_i)[\gamma_{00}e^{-H(\mathbf{x}_i)}-\gamma_{01}e^{H(\mathbf{x}_i)}]\Big\} \\
&+2\sinh(\alpha)\times \\
&\sum_{i=1}^{n}\Big\{\mathbb{1}(\tilde{y}_i=-1)\mathbb{1}(h(\mathbf{x}_i)\neq\tilde{y}_i)[\gamma_{11}e^{H(\mathbf{x}_i)}-\gamma_{10}e^{-H(\mathbf{x}_i)}]\Big\} \\
&+e^{-\alpha}\sum_{i=1}^{n}\Big\{\mathbb{1}(\tilde{y}_i=1)\gamma_{00}e^{-H(\mathbf{x}_i)}+\mathbb{1}(\tilde{y}_i=-1)\gamma_{11}e^{H(\mathbf{x}_i)}\Big\} \\
&+e^{\alpha}\sum_{i=1}^{n}\Big\{\mathbb{1}(\tilde{y}_i=1)\gamma_{01}e^{H(\mathbf{x}_i)}+\mathbb{1}(\tilde{y}_i=-1)\gamma_{10}e^{-H(\mathbf{x}_i)}\Big\}
\end{aligned} \tag{34}$$

Define $w_{00}=\gamma_{00}e^{-H(\mathbf{x}_i)}$, $w_{01}=\gamma_{01}e^{H(\mathbf{x}_i)}$, $w_{11}=\gamma_{11}e^{H(\mathbf{x}_i)}$ and $w_{10}=\gamma_{10}e^{-H(\mathbf{x}_i)}$, we simplify the objective into.

$$\begin{aligned}
\arg\min_{h,\alpha}\ & 2\sinh(\alpha)\sum_{i=1}^{n}\Big\{w_i\mathbb{1}(h(\mathbf{x}_i)\neq\tilde{y}_i)\Big\} \\
&+e^{-\alpha}\sum_{i=1}^{n}\Big\{\mathbb{1}(\tilde{y}_i=1)w_{00}+\mathbb{1}(\tilde{y}_i=-1)w_{11}\Big\} \\
&+e^{\alpha}\sum_{i=1}^{n}\Big\{\mathbb{1}(\tilde{y}_i=1)w_{01}+\mathbb{1}(\tilde{y}_i=-1)w_{10}\Big\}
\end{aligned} \tag{35}$$

where

$$w_i=\begin{cases}(w_{00}-w_{01}), & \text{if } \tilde{y}_i=+1.\\ (w_{11}-w_{10}), & \text{if } \tilde{y}_i=-1.\end{cases} \tag{36}$$

References

Barandela, Ricardo and Gasca, Eduardo. Decontamination of Training Samples for Supervised Pattern Recognition Methods. In *Advances in Pattern Recognition*, volume 1876 of *Lecture Notes in Computer Science*, pp. 621–630. 2000.

Bootkrajang, Jakramate and Kabán, Ata. Label-Noise Robust Logistic Regression and Its Applications. In *Proceedings of the 2012 European conference on Machine learning and knowledge discovery in databases - Volume Part I*, ECML-PKDD'12, pp. 143–158. Springer-Verlag, 2012.

Bootkrajang, Jakramate and Kabán, Ata. Classification of mislabelled microarrays using robust sparse logistic regression. *Bioinformatics*, 29(7):870–877, 2013.

Bouveyron, Charles and Girard, Stephane. Robust supervised classification with mixture models: Learning from data with uncertain labels. *Pattern Recognition*, 42(11):2649–2658, 2009.

Fan, Wei and Stolfo, Salvatore J. AdaCost: misclassification cost-sensitive boosting. In *Proeedings of the 16th International Conference on Machine Learning*, pp. 97–105. Morgan Kaufmann, 1999.

Frank, J. Andrew. and Asuncion, Arthur. UCI Machine Learning Repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Freund, Yoav. A more robust boosting algorithm. Technical Report arXiv:0905.2138, 2009.

Friedman, Jerome, Hastie, Trevor, and Tibshirani, Robert. Additive Logistic Regression: a Statistical View of Boosting. *Annals of Statistics*, 28, 1998.

Hernández-Lobato, Daniel, Hernández-Lobato, José Miguel, and Dupont, Pierre. Robust Multi-Class Gaussian Process Classification. In *NIPS*, pp. 280–288, 2011.

Karmaker, Amitava and Kwek, Stephen. A boosting approach to remove class label noise. *International Journal of Hybrid Intelligent Systems*, 3(3):169–177, August 2006.

Krieger, Abba, Long, Chuan, and Wyner, Abraham. Boosting Noisy Data. In *Proceedings of the 18th International Conference on Machine Learning*, ICML'01, pp. 274–281. Morgan Kaufmann, 2001.

Lawrence, Neil D. and Schölkopf, Bernhard. Estimating a Kernel Fisher Discriminant in the Presence of Label Noise. In *Proceedings of the 18th International Conference on Machine Learning*, pp. 306–313. Morgan Kaufmann, 2001.

Long, Philip M. and Servedio, Rocco A. Random classification noise defeats all convex potential boosters. *Machine Learning*, 78(3):287–304, March 2010.

Masnadi-Shirazi, Hamed and Vasconcelos, Nuno. Cost-sensitive boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):294–309, 2011.

Niculescu-Mizil, Alexandru and Caruana, Rich. Obtaining Calibrated Probabilities from Boosting. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, UAI'05, pp. 413–420. AUAI Press, 2005.

Platt, John C. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *Advances in large margin classifiers*, pp. 61–74. MIT Press, 1999.

Raykar, Vikas C., Yu, Shipeng, Zhao, Linda H., Valadez, Gerardo Hermosillo, Florin, Charles, Bogoni, Luca, and Moy, Linda. Learning From Crowds. *Journal of Machine Learning Research*, 11:1297–1322, 2010.

Robertson, Tim. Wright F. T. Dykstra Richard L. *Order Restricted Statistical Inference.* John Wiley and Sons, New York, 1988.

Takenouchi, Takashi, Eguchi, Shinto, Murata, Noboru, and Kanamori, Takafumi. Robust boosting algorithm against mislabeling in multiclass problems. *Neural Computation*, 20(6):1596–1630, June 2008.

Vezhnevets, Alexander and Vezhnevets, Vladimir. Modest AdaBoost' – Teaching AdaBoost to Generalize Better. In *GraphiCon*, Novosibirsk Akademgorodok, Russia, 2005.

Hilbert Space Embeddings of Predictive State Representations

Byron Boots
Computer Science and Engineering Dept.
University of Washington
Seattle, WA

Arthur Gretton
Gatsby Unit
University College London
London, UK

Geoffrey J. Gordon
Machine Learning Dept.
Carnegie Mellon University
Pittsburgh, PA

Abstract

Predictive State Representations (PSRs) are an expressive class of models for controlled stochastic processes. PSRs represent state as a set of predictions of future observable events. Because PSRs are defined entirely in terms of observable data, statistically consistent estimates of PSR parameters can be learned efficiently by manipulating moments of observed training data. Most learning algorithms for PSRs have assumed that actions and observations are finite with low cardinality. In this paper, we generalize PSRs to infinite sets of observations and actions, using the recent concept of Hilbert space embeddings of distributions. The essence is to represent the state as one or more nonparametric *conditional embedding operators* in a Reproducing Kernel Hilbert Space (RKHS) and leverage recent work in kernel methods to estimate, predict, and update the representation. We show that these *Hilbert space embeddings of PSRs* are able to gracefully handle continuous actions and observations, and that our learned models outperform competing system identification algorithms on several prediction benchmarks.

1 INTRODUCTION

Many problems in machine learning and artificial intelligence involve discrete-time partially observable nonlinear dynamical systems. If the observations are discrete, then *Hidden Markov Models* (HMMs) [1] or, in the control setting, *Input-Output HMMs* (IO-HMMs) [2], can be used to represent belief as a discrete distribution over latent states. *Predictive State Representations (PSRs)* [3] are generalizations of IO-HMMs that have attracted interest because they can have greater representational capacity for a fixed model dimension. In contrast to latent-variable representations like HMMs, PSRs represent the state of a dynamical system by tracking occurrence probabilities of future observable events (called *tests*) conditioned on past observable events (called *histories*). One of the prime motivations for modeling dynamical systems with PSRs was that, because tests and histories are observable quantities, learning PSRs should be easier than learning IO-HMMs by heuristics like Expectation Maximization (EM), which suffer from bad local optima and slow convergence rates.

For example, Boots et al. [4] proposed a *spectral* algorithm for learning PSRs with discrete observations and actions. At its core, the algorithm performs a singular value decomposition of a matrix of joint probabilities of tests and partitions of histories (the moments mentioned above), and then uses linear algebra to recover parameters that allow predicting, simulating, and filtering in the modeled system. As hinted above, the algorithm is statistically consistent, and does not need to resort to local search—an important benefit compared to typical heuristics (like EM) for learning latent variable representations.

Despite their positive properties, many algorithms for PSRs are restricted to discrete observations and actions with only moderate cardinality. For continuous actions and observations, and for actions and observations with large cardinalities, learning algorithms for PSRs often run into trouble: we cannot hope to see each action or observation more than a small number of times, so we cannot gather enough data to estimate the PSR parameters accurately without additional assumptions. Previous approaches attempt to learn continuous PSRs by leveraging kernel density estimation [4] or modeling PSR distributions with exponential families [5, 6]; each of these methods must contend with drawbacks such as slow rates of statistical convergence and difficult numerical integration.

In this paper, we fully generalize PSRs to continuous observations and actions using a recent concept called Hilbert space embeddings of distributions [7, 8].

The essence of our method is to represent distributions of tests, histories, observations, and actions as points in (possibly) infinite-dimensional reproducing kernel Hilbert spaces. During filtering we update these embedded distributions using a kernel version of Bayes' rule [9]. The advantage of this approach is that embedded distributions can be estimated accurately without having to contend with problems such as density estimation and numerical integration. Depending on the kernel, the model can be parametric or nonparametric. We focus on the nonparametric case: we leverage the "kernel trick" to represent the state and required operators implicitly and maintain a state vector with length proportional to the size of the training dataset.

1.1 RELATED WORK

Our approach is similar to recent work that applies kernel methods to dynamical system modeling and reinforcement learning, which we summarize here. Song et al. [10] proposed a nonparametric approach to learning HMM representations in RKHSs. The resulting dynamical system model, called Hilbert Space Embeddings of Hidden Markov Models (HSE-HMMs), proved to be more accurate compared to competing models on several experimental benchmarks [10, 11]. Despite these successes, HSE-HMMs have two major limitations: first, the update rule for the HMM relies on *density estimation* instead of Bayesian inference in Hilbert space, which results in an awkward model with poor theoretical guarantees. Second, the model lacks the capacity to reason about actions, which limits the scope of the algorithm. Our model can be viewed as an extension of HSE-HMMs that adds inputs and updates state using a kernelized version of Bayes' rule.

Grünewälder et al. [12] proposed a nonparametric approach to learning transition dynamics in Markov decision processes (MDPs) by representing the stochastic transitions as conditional distributions in RKHS. This work was extended to POMDPs by Nishiyama et al. [13]. Like the approach we propose here, the resulting Hilbert space embedding of POMDPs represents distributions over the states, observations, and actions as embeddings in RKHS and uses kernel Bayes' rule to update these distribution embeddings. Critically, the algorithm requires training data that includes labels for the true latent states. This is a serious limitation: it precludes learning dynamical systems directly from sensory data. By contrast, our algorithm only requires access to an unlabeled sequence of actions and observations, and learns the more expressive PSR model, which includes POMDPs as a special case.

2 PSRS

A PSR represents the state of a controlled stochastic process as a set of predictions of observable experiments or *tests* that can be performed in the system. Specifically, a test of length N is an ordered sequence of future action-observations pairs $\tau = a_1, o_1, \ldots a_N, o_N$ that can be selected and observed at any time t. Likewise, a *history* is an ordered sequence of actions and observations $h = a_1, o_1, \ldots, a_M, o_M$ that have been selected and observed prior to t.

A test τ_i is *executed* at time t if we intervene [14] to select the sequence of actions specified by the test $\tau_i^{\mathcal{A}} = a_1, \ldots, a_N$. It is said to *succeed* at time t if it is executed and the sequence of observations in the test $\tau_i^{\mathcal{O}} = o_1, \ldots, o_N$ matches the observations generated by the system. The *prediction* for test i at time t is the probability of the test succeeding given a history h_t and given that we execute it:[1]

$$\mathbb{P}\left[\tau_{i,t}^{\mathcal{O}} \mid \tau_{i,t}^{\mathcal{A}}, h_t\right] = \frac{\mathbb{P}\left[\tau_i^{\mathcal{O}}, \tau_i^{\mathcal{A}} \mid h_t\right]}{\mathbb{P}\left[\tau_i^{\mathcal{A}} \mid h_t\right]} \tag{1}$$

The key idea behind a PSR is that if we know the expected outcomes of executing *all* possible tests, then we know everything there is to know about the state of a dynamical system [16]. In practice we will work with the predictions of some *set* of tests; therefore, let $\mathcal{T} = \{\tau_i\}$ be a set of d tests. We write

$$s(h_t) = \left(\mathbb{P}\left[\tau_{i,t}^{\mathcal{O}} \mid \tau_{i,t}^{\mathcal{A}}, h_t\right]\right)_{i=1}^{d} \tag{2}$$

for the *prediction vector* of success probabilities for the tests $\tau_i \in \mathcal{T}$ given a history h_t.

Knowing the success probabilities of some tests may allow us to compute the success probabilities of other tests. That is, given a test τ_l and a prediction vector $s(h_t)$, there may exist a *prediction function* f_{τ_l} such that $\mathbb{P}\left[\tau_l^{\mathcal{O}} \mid \tau_l^{\mathcal{A}}, h_t\right] = f_{\tau_l}(s(h_t))$. In this case, we say $s(h_t)$ is a *sufficient statistic* for $\mathbb{P}\left[\tau_l^{\mathcal{O}} \mid \tau_l^{\mathcal{A}}, h_t\right]$. A *core* set of tests is a set whose prediction vector $s(h_t)$ is a sufficient statistic for the predictions of *all* tests τ_l at time t. Therefore, $s(h_t)$ is a *state* for our PSR: i.e., at each time step t we can remember $s(h_t)$ instead h_t.

Formally, a PSR is a tuple $\langle \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{F}, s_o \rangle$. $\mathcal{O}$ is the set of possible observations and $\mathcal{A}$ is the set of possible actions. $\mathcal{T}$ is a core set of tests. $\mathcal{F}$ is the set of prediction functions f_{τ_l} for *all* tests τ_l (which must exist since $\mathcal{T}$ is a core set), and $s_0 = s(h_0)$ is the initial prediction vector after seeing the empty history h_0.

In this paper we restrict ourselves to *linear* PSRs, in which all prediction functions are linear: $f_{\tau_l}(s(h_t)) =$

[1]For simplicity, we assume that all probabilities involving actions refer to our PSR as controlled by an arbitrary *blind* or *open-loop* policy [15] (also called *exogenous inputs*). In this case, conditioning on $\mathrm{do}(a_1, \ldots, a_M)$ is equivalent to conditioning on observing $a_1, \ldots, a_M$, which allows us to avoid some complex notation and derivations.

$f_{\tau_l}^{\mathsf{T}} s(h_t)$ for some vector $f_{\tau_l} \in \mathbb{R}^{|\mathcal{T}|}$. Note that the restriction to linear prediction functions is only a restriction to linear relationships between conditional probabilities of tests; linear PSRs can still represent systems with *nonlinear* dynamics.

2.1 FILTERING WITH BAYES' RULE

After taking action a and seeing observation o, we can update the state $s(h_t)$ to the state $s(h_{t+1})$ by Bayes' rule. The key idea is that the set of functions $\mathcal{F}$ allows us to predict *any* test from our core set of tests.

The state update proceeds as follows: first, we predict the success of any core test τ_i prepended by an action a and an observation o, which we call $ao\tau_i$, as a linear function of our core test predictions $s(h_t)$:

$$\mathbb{P}\left[\tau_{i,t+1}^{\mathcal{O}}, o_t{=}o \mid \tau_{i,t+1}^{\mathcal{A}}, a_t{=}a, h_t\right] = f_{ao\tau_i}^{\mathsf{T}} s(h_t) \tag{3}$$

Second, we predict the likelihood of any observation o given that we select action a (i.e., the test ao):

$$\mathbb{P}\left[o_t = o \mid a_t = a, h_t\right] = f_{ao}^{\mathsf{T}} s(h_t) \tag{4}$$

After executing action a and seeing observation o, Equations 3 and 4 allow us to find the prediction for a core test τ_i from $s(h_t)$ using Bayes' Rule:

$$\begin{aligned} s_i(h_{t+1}) &= \mathbb{P}\left[\tau_{i,t+1}^{\mathcal{O}} \mid \tau_{i,t+1}^{\mathcal{A}}, a_t = a, o_t = o, h_t\right] \\ &= \frac{\mathbb{P}\left[\tau_{i,t+1}^{\mathcal{O}}, o_t = o \mid \tau_{i,t+1}^{\mathcal{A}}, a_t = a, h_t\right]}{\mathbb{P}\left[o_t = o \mid a_t = a, h_t\right]} \\ &= \frac{f_{ao\tau_i}^{\mathsf{T}} s(h_t)}{f_{ao}^{\mathsf{T}} s(h_t)} \end{aligned} \tag{5}$$

This recursive application of Bayes' rule to a belief state is called a *Bayes filter*.

3 HILBERT SPACE EMBEDDINGS

The key idea in this paper is to represent (possibly continuous) distributions of tests, histories, observations, and actions *nonparametrically* as points in (possibly infinite dimensional) Hilbert spaces. During filtering these points are updated entirely in Hilbert space, mirroring the finite-dimensional updates, using a kernel version of Bayes' rule.

3.1 MEAN MAPS

Let $\mathcal{F}$ be a reproducing kernel Hilbert space (RKHS) associated with kernel $K_X(x, x') \overset{\text{def}}{=} \left\langle \phi^X(x), \phi^X(x') \right\rangle_{\mathcal{F}}$ for $x \in \mathcal{X}$. Let $\mathcal{P}$ be the set of probability distributions on $\mathcal{X}$, and X be a random variable with distribution $\mathbb{P} \in \mathcal{P}$. Following Smola et al. [7], we define the mean map (or the embedding) of $\mathbb{P} \in \mathcal{P}$ into RKHS $\mathcal{F}$ to be $\mu_X \overset{\text{def}}{=} \mathbb{E}\left[\phi^X(X)\right]$.

A *characteristic* RKHS is one for which the mean map is injective: that is, each distribution $\mathbb{P}$ has a unique embedding [8]. This property holds for many commonly used kernels, e.g., the Gaussian and Laplace kernels when $\mathcal{X} = \mathbb{R}^d$.

Given *i.i.d.* observations x_t, $t = 1 \ldots T$, an estimate of the mean map is straightforward:

$$\hat{\mu}_X \overset{\text{def}}{=} \frac{1}{T} \sum_{t=1}^{T} \phi^X(x_t) = \frac{1}{T} \Upsilon^X \mathbf{1}_T \tag{6}$$

where $\Upsilon^X \overset{\text{def}}{=} (\phi^X(x_1), \ldots, \phi^X(x_T))$ is the linear operator which maps the tth unit vector of $\mathbb{R}^T$ to $\phi^X(x_t)$.

Below, we'll sometimes need to embed a joint distribution $\mathbb{P}[X, Y]$. It is natural to embed $\mathbb{P}[X, Y]$ into a tensor product RKHS: let $K_Y(y, y') = \left\langle \phi^Y(y), \phi^Y(y') \right\rangle_{\mathcal{G}}$ be a kernel on $\mathcal{Y}$ with associated RKHS $\mathcal{G}$. Then we write μ_{XY} for the mean map of $\mathbb{P}[X, Y]$ under the kernel $K_{XY}((x, y), (x', y')) \overset{\text{def}}{=} K_X(x, x') K_Y(y, y')$ for the tensor product RKHS $\mathcal{F} \otimes \mathcal{G}$.

3.2 COVARIANCE OPERATORS

The covariance operator is a generalization of the covariance matrix. Given a joint distribution $\mathbb{P}[X, Y]$ over two variables X on $\mathcal{X}$ and Y on $\mathcal{Y}$, the *uncentered* covariance operator $\mathcal{C}_{XY}$ is the linear operator which satisfies [17]

$$\langle f, \mathcal{C}_{XY} g \rangle_{\mathcal{F}} = \mathbb{E}_{XY}\left[f(X) g(Y)\right] \quad \forall f \in \mathcal{F}, g \in \mathcal{G} \tag{7}$$

Both μ_{XY} and $\mathcal{C}_{XY}$ represent the distribution $\mathbb{P}[X, Y]$. One is defined as an element of $\mathcal{F} \otimes \mathcal{G}$, and the other as a linear operator from $\mathcal{G}$ to $\mathcal{F}$, but they are isomorphic under the standard identification of these spaces [9], so we abuse notation and write $\mu_{XY} = \mathcal{C}_{XY}$.

Given T *i.i.d.* pairs of observations (x_t, y_t), define $\Upsilon^X = \left(\phi^X(x_1), \ldots, \phi^X(x_T)\right)$ and $\Upsilon^Y = \left(\phi^Y(y_1), \ldots, \phi^Y(y_T)\right)$. Write Υ^* for the adjoint of Υ. Analogous to (6), we can estimate

$$\widehat{\mathcal{C}}_{XY} = \frac{1}{T} \Upsilon^X \Upsilon^{Y*} \tag{8}$$

3.3 CONDITIONAL OPERATORS

Based on covariance operators, Song et al. [18] define a linear operator $\mathcal{W}_{Y|X} : \mathcal{F} \mapsto \mathcal{G}$ that allows us to compute conditional expectations $\mathbb{E}\left[\phi^Y(Y) \mid x\right]$ in RKHSs. Given some smoothness assumptions [18], this conditional embedding operator is

$$\mathcal{W}_{Y|X} \overset{\text{def}}{=} \mathcal{C}_{YX} \mathcal{C}_{XX}^{-1} \tag{9}$$

and for all $g \in \mathcal{G}$ we have

$$\mathbb{E}[g(Y) \mid x] = \langle g, \mathcal{W}_{Y|X} \phi^X(x) \rangle_{\mathcal{G}}$$

Given T *i.i.d.* pairs (x_t, y_t) from $\mathbb{P}[X, Y]$, we can estimate $\mathcal{W}_{Y|X}$ by kernel ridge regression [18, 19]:

$$\widehat{\mathcal{W}}_{Y|X} = (1/T) \Upsilon^Y \left((1/T) \Upsilon^X\right)_{\lambda}^{\dagger}$$

where the regularized pseudoinverse $\Upsilon_\lambda^\dagger$ is given by $\Upsilon_\lambda^\dagger = \Upsilon^*(\Upsilon\Upsilon^* + \lambda I)^{-1}$. (The regularization parameter λ helps to avoid overfitting and to ensure invertibility, and thus that the resulting operator is well defined.) Equivalently,

$$\widehat{\mathcal{W}}_{Y|X} = \Upsilon^Y (G_{X,X} + \lambda T I)^{-1} \Upsilon^{X^*}$$

where the Gram matrix $G_{X,X} \overset{\text{def}}{=} \Upsilon^{X^*}\Upsilon^X$ has (i,j)th entry $K_X(x_i, x_j)$.

3.4 KERNEL BAYES' RULE

We are now in a position to define the kernel mean map implementation of Bayes' rule (called the Kernel Bayes' Rule, or KBR). In particular, we want the kernel analog of $\mathbb{P}[X \mid y, z] = \mathbb{P}[X, y \mid z] / \mathbb{P}[y \mid z]$. In deriving the kernel realization of this rule we need the kernel mean representation of a conditional *joint* probability $\mathbb{P}[X, Y \mid z]$. Given Hilbert spaces $\mathcal{F}$, $\mathcal{G}$, and $\mathcal{H}$ corresponding to the random variables X, Y, and Z respectively, $\mathbb{P}[X, Y \mid z]$ can be represented as a mean map $\mu_{XY|z} \overset{\text{def}}{=} \mathbb{E}\left[\phi^X(X) \otimes \phi^Y(Y) \mid z\right]$ or the corresponding operator $\mathcal{C}_{XY|z}$. Under some assumptions [9], and with a similar abuse of notation as before, this operator satisfies:

$$\mathcal{C}_{XY|z} = \mu_{XY|z} \overset{\text{def}}{=} \mathcal{C}_{(XY)Z}\mathcal{C}_{ZZ}^{-1}\phi(z) \tag{10}$$

Here the operator $\mathcal{C}_{(XY)Z}$ represents the covariance of the random variable (X, Y) with the random variable Z. (We can view (10) as applying a conditional embedding operator $\mathcal{W}_{XY|Z}$ to an observation z.) We now define KBR in terms of conditional covariance operators [9]:

$$\mu_{X|y,z} = \mathcal{C}_{XY|z}\mathcal{C}_{YY|z}^{-1}\phi(y) \tag{11}$$

To map the KBR to the ordinary Bayes' rule above, $\mu_{X|y,z}$ is the embedding of $\mathbb{P}[X \mid y, z]$; $\mathcal{C}_{XY|z}$ is the embedding of $\mathbb{P}[X, Y \mid z]$; and the action of $\mathcal{C}_{YY|z}^{-1}\phi(y)$ corresponds to substituting $Y = y$ into $\mathbb{P}[X, Y \mid z]$ and dividing by $\mathbb{P}[y \mid z]$.

To use KBR in practice, we need to estimate the operators on the RKHS of (11) from data. Given T *i.i.d.* triples (x_t, y_t, z_t) from $\mathbb{P}[X, Y, Z]$, write $\Upsilon^X = \left(\phi^X(x_1), \ldots, \phi^X(x_T)\right)$, $\Upsilon^Y = \left(\phi^Y(y_1), \ldots, \phi^Y(y_T)\right)$, and $\Upsilon^Z = \left(\phi^Z(z_1), \ldots, \phi^Z(z_T)\right)$. We can now estimate the covariance operators $\widehat{\mathcal{C}}_{XY|z}$ and $\widehat{\mathcal{C}}_{YY|z}$ via Equation 10; applying KBR, we get $\widehat{\mathcal{C}}_{X|y,z} = \widehat{\mathcal{C}}_{XY|z}\left(\widehat{\mathcal{C}}_{YY|z} + \lambda I\right)^{-1}\phi^Y(y)$. We express this process with Gram matrices, using a ridge parameter λ that goes to zero at an appropriate rate with T [9]:

$$\Lambda_z = \operatorname{diag}((G_{Z,Z} + \lambda T I)^{-1}\Upsilon^{Z^*}\phi^Z(z)) \tag{12}$$

$$\widehat{\mathcal{W}}_{X|Y,z} = \Upsilon^X(\Lambda_z G_{Y,Y} + \lambda T I)^{-1}\Lambda_z\Upsilon^{Y^*} \tag{13}$$

$$\widehat{\mu}_{X|y,z} = \widehat{\mathcal{W}}_{X|Y,z}\phi^Y(y) \tag{14}$$

where $G_{Y,Y} \overset{\text{def}}{=} \Upsilon^{Y^*}\Upsilon^Y$ has (i,j)th entry $K_Y(y_i, y_j)$, and $G_{Z,Z} \overset{\text{def}}{=} \Upsilon^{Z^*}\Upsilon^Z$ has (i,j)th entry $K_Z(z_i, z_j)$. The diagonal elements of Λ_z weight the samples, encoding the conditioning information from z.

4 RKHS EMBEDDINGS OF PSRS

We are now ready to apply Hilbert space embeddings to PSRs. For now we ignore the question of learning, and simply suppose that we are given representations of the RKHS operators described below. In Section 4.1 we show how predictive states can be represented as mean embeddings. In Section 4.2 we generalize the notion of a core set of tests and define the Hilbert space embedding of PSRs. Finally, in Section 4.3 we show how to perform filtering in our embedded PSR with Kernel Bayes' Rule. We return to learning in Section 5.

4.1 PREDICTIVE STATE EMBEDDINGS

We begin by defining kernels on length-N sequences of test observations $\tau^{\mathcal{O}}$, test actions $\tau^{\mathcal{A}}$, and histories h: $K_{\mathcal{T}^{\mathcal{O}}}(\tau^{\mathcal{O}}, \tau'^{\mathcal{O}}) \overset{\text{def}}{=} \langle\phi^{\mathcal{T}^{\mathcal{O}}}(\tau^{\mathcal{O}}), \phi^{\mathcal{T}^{\mathcal{O}}}(\tau'^{\mathcal{O}})\rangle_{\mathcal{F}}$, $K_{\mathcal{T}^{\mathcal{A}}}(\tau^{\mathcal{A}}, \tau'^{\mathcal{A}}) \overset{\text{def}}{=} \langle\phi^{\mathcal{T}^{\mathcal{A}}}(\tau^{\mathcal{A}}), \phi^{\mathcal{T}^{\mathcal{A}}}(\tau'^{\mathcal{A}})\rangle_{\mathcal{G}}$, and $K_{\mathcal{H}}(h, h') \overset{\text{def}}{=} \left\langle\phi^{\mathcal{H}}(h), \phi^{\mathcal{H}}(h')\right\rangle_{\mathcal{L}}$. Define also the mean maps

$$\mu_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}},\mathcal{H}} \overset{\text{def}}{=} \mathbb{E}\left[\phi^{\mathcal{T}^{\mathcal{A}}}(\tau^{\mathcal{A}}) \otimes \phi^{\mathcal{T}^{\mathcal{A}}}(\tau^{\mathcal{A}}) \otimes \phi^{\mathcal{H}}(\mathcal{H}_t)\right] \tag{15}$$

$$\mu_{\mathcal{T}^{\mathcal{O}},\mathcal{T}^{\mathcal{A}},\mathcal{H}} \overset{\text{def}}{=} \mathbb{E}\left[\phi^{\mathcal{T}^{\mathcal{O}}}(\tau^{\mathcal{O}}) \otimes \phi^{\mathcal{T}^{\mathcal{A}}}(\tau^{\mathcal{A}}) \otimes \phi^{\mathcal{H}}(\mathcal{H}_t)\right] \tag{16}$$

which correspond to operators $\mathcal{C}_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}},\mathcal{H}}$ and $\mathcal{C}_{\mathcal{T}^{\mathcal{O}},\mathcal{T}^{\mathcal{A}},\mathcal{H}}$. We now take our PSR state to be the conditional embedding operator which predicts test observations from test actions:

$$\mathcal{S}(h_t) = \mathcal{W}_{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}},h_t} = \mathcal{C}_{\mathcal{T}^{\mathcal{O}},\mathcal{T}^{\mathcal{A}}|h_t}\mathcal{C}_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}}|h_t}^{-1} \tag{17}$$

where $\mathcal{C}_{\mathcal{T}^{\mathcal{O}},\mathcal{T}^{\mathcal{A}}|h_t} = \mathcal{C}_{\mathcal{T}^{\mathcal{O}},\mathcal{T}^{\mathcal{A}}\mathcal{H}}\mathcal{C}_{\mathcal{H},\mathcal{H}}^{-1}\phi^{\mathcal{H}}(h_t)$ and $\mathcal{C}_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}}|h_t} = \mathcal{C}_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}},\mathcal{H}}\mathcal{C}_{\mathcal{H},\mathcal{H}}^{-1}\phi^{\mathcal{H}}(h_t)$. This definition is analogous to the finite-dimensional case, in which the PSR state is a conditional probability table instead of a conditional embedding operator.[2]

Given characteristic RKHSs, the operator $\mathcal{S}(h_t)$ uniquely encodes the predictive densities of future observation sequences given that we take future action sequences. This is an expressive representation: we can model near-arbitrary continuous-valued distributions, limited only by the existence of the conditional

[2]In contrast to discrete PSRs, we typically consider the entire set of length-N tests at once; this change makes notation simpler, and is no loss of generality since the embedding includes the information needed to predict any individual test of length up to N. (Computationally, we always work with sample-based representations, so the size of our set of tests doesn't matter.)

embedding operator $\mathcal{W}_{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}},h_t}$ (and therefore the assumptions in Section 3.3).

4.2 CORE TESTS AND HSE-PSRS

As defined above, the embedding $\mathcal{S}(h_t)$ lets us compute predictions for a special set of tests, namely length-N futures. As with discrete PSRs, knowing the predictions for some tests may allow us to compute the predictions for other tests. For example, given the embedding $\mathcal{S}(h_t)$ and another set of tests $\mathcal{T}$, there may exist a function $\mathcal{F}_{\mathcal{T}}$ such the predictions for $\mathcal{T}$ can be computed as $\mathcal{W}_{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}},h_t} = \mathcal{F}_{\mathcal{T}}(\mathcal{S}(h_t))$. In this case, $\mathcal{S}(h_t)$ is a *sufficient statistic* for $\mathcal{T}$. Here, as with discrete PSRs, we focus on prediction functions that are *linear* operators; however, this assumption is mild compared to the finite case, since linear operators on infinite-dimensional RKHSs are very expressive.

A *core* set of tests is defined similarly to the discrete PSR case (Section 2): a core set is one whose embedding $\mathcal{S}(h_t)$ is a linearly sufficient statistic for the prediction of distribution embeddings of *any* finite length. Therefore, $\mathcal{S}(h_t)$ is a *state* for an embedded PSR: at each time step t we remember the embedding of test predictions $\mathcal{S}(h_t)$ instead of h_t.

Formally, a *Hilbert space embedding of a PSR* (HSE-PSR) is a tuple $\langle K_{\mathcal{O}}(o,o'), K_{\mathcal{A}}(a,a'), N, \mathcal{F}, \mathcal{S}_o\rangle$. $K_{\mathcal{O}}(o,o')$ is a characteristic kernel on observations and $K_{\mathcal{A}}(a,a')$ is a characteristic kernel on actions. N is a positive integer such that the set of length-N tests is core. $\mathcal{F}$ is the set of linear operators for predicting embeddings of any-length test predictions from the length-N embedding (which must exist since length-N tests are a core set), and $\mathcal{S}_0 = \mathcal{S}(h_0)$ is the initial prediction for our core tests given the null history h_0.

4.3 UPDATING STATE WITH KERNEL BAYES' RULE

Given an action a and an observation o, the HSE-PSR state update is computed using the kernel versions of conditioning and Bayes rule given in Section 3. As in Section 2, the key idea is that the set of functions $\mathcal{F}$ allows us to predict the embedding of the predictive distribution of *any* sequence of observations from the embedding of our core set of tests $\mathcal{S}(h_t)$.

The first step in updating the state is finding the embedding of tests of length $N+1$. By our assumptions, a linear operator $\mathcal{F}_{\mathcal{AOT}}$ exists which accomplishes this:

$$\mathcal{W}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A},h_t} = \mathcal{F}_{\mathcal{AOT}}\mathcal{S}(h_t) \tag{18}$$

The second step is finding the embedding of observation likelihoods at time t given actions. By our assumptions, we can do so with an operator $\mathcal{F}_{\mathcal{AO}}$:

$$\mathcal{W}_{\mathcal{O},\mathcal{O}|\mathcal{A},h_t} = \mathcal{F}_{\mathcal{AO}}\mathcal{S}(h_t) \tag{19}$$

With the two embeddings $\mathcal{W}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A},h_t}$ and $\mathcal{W}_{\mathcal{O},\mathcal{O}|\mathcal{A},h_t}$, we can update the state given a new action and observation. First, when we choose an action a_t, we compute the conditional embeddings:

$$\mathcal{C}_{\mathcal{O},\mathcal{O}|h_t,a_t} = \mu_{\mathcal{O},\mathcal{O}|h_t,a_t} = \mathcal{W}_{\mathcal{O},\mathcal{O}|\mathcal{A},h_t}\phi^{\mathcal{A}}(a_t) \tag{20}$$

$$\mathcal{W}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},h_t,a_t} = \mathcal{W}_{\mathcal{T}^{\mathcal{O}'}\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A},h_t} \times_{\mathcal{A}} \phi^{\mathcal{A}}(a_t) \tag{21}$$

Here, $\times_{\mathcal{A}}$ specifies that we are thinking of $\mathcal{W}_{\mathcal{T}^{\mathcal{O}'}\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A},h_t}$ as a tensor with 4 modes, one for each of $\mathcal{T}^{\mathcal{O}'}$, $\mathcal{O}$, $\mathcal{T}^{\mathcal{A}'}$, $\mathcal{A}$, and contracting along the mode $\mathcal{A}$ corresponding to the current action. Finally, when we receive the observation o_t, we calculate the next state by KBR:

$$\mathcal{S}(h_{t+1}) \equiv \mathcal{W}_{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'},h_t,a_t,o_t} = \mathcal{W}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},h_t,a_t} \times_{\mathcal{O}} \mathcal{C}^{-1}_{\mathcal{O},\mathcal{O}|h_t,a_t}\phi^{\mathcal{O}}(o_t) \tag{22}$$

Here, $\times_{\mathcal{O}}$ specifies that we are thinking of $\mathcal{W}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},h_t,a_t}$ as a tensor with 3 modes and contracting along the mode corresponding to the current observation.

5 LEARNING HSE-PSRS

If the RKHS embeddings are *finite* and low-dimensional, then the learning algorithm and state update are straightforward: we estimate the conditional embedding operators directly, learn the functions $\mathcal{F}_{\mathcal{AOT}}$ and $\mathcal{F}_{\mathcal{AO}}$ by linear regression, and update our state with Bayes' rule via Eqs. 18–22. See, for example [4] or [20]. However, if the RKHS is *infinite*, e.g., if we use Gaussian RBF kernels, then it is not possible to store or manipulate HSE-PSR state directly. In Sections 5.1–5.3, we show how learn a HSE-PSR in potentially-infinite RKHSs by leveraging the "kernel trick" and Gram matrices to represent all of the required operators implicitly. Section 5.1 describes how to represent HSE-PSR states as vectors of weights on sample histories; Section 5.2 describes how to learn the operators needed for updating states; and Section 5.3 describes how to update the state weights recursively using these operators.

5.1 A GRAM MATRIX FORMULATION

5.1.1 The HSE-PSR State

We begin by describing how to represent the HSE-PSR state in Eq. 17 as a weighted combination of training data samples. Given T *i.i.d.* tuples $\left\{(\tau_t^{\mathcal{O}}, \tau_t^{\mathcal{A}}, h_t)\right\}_{t=1}^{T}$ generated by a stochastic process controlled by a blind policy, we denote:[3]

[3]To get independent samples, we'd need to reset our process between samples, or run it long enough that it mixes. In practice we can use dependent samples (as we'd get from a single long trace) at the cost of reducing the convergence rate in proportion to the mixing time. We can also use dependent samples in Sec. 5.1.2 due to our careful choice of which operators to estimate.

$$\Upsilon^{\mathcal{T}^{\mathcal{O}}} = \left(\phi^{\mathcal{T}^{\mathcal{O}}}(\tau_1^{\mathcal{O}}), \ldots, \phi^{\mathcal{T}^{\mathcal{O}}}(\tau_T^{\mathcal{O}})\right) \tag{23}$$

$$\Upsilon^{\mathcal{T}^{\mathcal{A}}} = \left(\phi^{\mathcal{T}^{\mathcal{A}}}(\tau_1^{\mathcal{A}}), \ldots, \phi^{\mathcal{T}^{\mathcal{A}}}(\tau_T^{\mathcal{A}})\right) \tag{24}$$

$$\Upsilon^{\mathcal{H}} = \left(\phi^{\mathcal{H}}(h_1), \ldots, \phi^{\mathcal{H}}(h_T)\right) \tag{25}$$

and define Gram matrices:

$$G_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}}} = \Upsilon^{\mathcal{T}^{\mathcal{A}}*}\Upsilon^{\mathcal{T}^{\mathcal{A}}} \tag{26}$$

$$G_{\mathcal{H},\mathcal{H}} = \Upsilon^{\mathcal{H}*}\Upsilon^{\mathcal{H}} \tag{27}$$

We can then calculate an estimate of the state at time t in our training sample (Eq. 17) using Eqs. 12 and 13 from the kernel Bayes' rule derivation:

$$\alpha_{h_t} = (G_{\mathcal{H},\mathcal{H}} + \lambda T I)^{-1}\Upsilon^{\mathcal{H}*}\phi^{\mathcal{H}}(h_t) \tag{28}$$

$$\Lambda_{h_t} = \operatorname{diag}(\alpha_{h_t}) \tag{29}$$

$$\widehat{\mathcal{S}}(h_t) = \Upsilon^{\mathcal{T}^{\mathcal{O}}}(\Lambda_{h_t}G_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}}} + \lambda T I)^{-1}\Lambda_{h_t}\Upsilon^{\mathcal{T}^{\mathcal{A}}*} \tag{30}$$

We will use these training set state estimates below to help learn state update operators for our HSE-PSR.

5.1.2 Vectorized States

The state update operators treat states as vectors (e.g., mapping a current state to an expected future state). The state in Eq. 30 is written as an operator, so to put it in the more-convenient vector form, we want to do the infinite-dimensional equivalent of reshaping a matrix to a vector. To see how, we can look at the example of the covariance operator $\widehat{C}_{\mathcal{T}^{\mathcal{O}},\mathcal{T}^{\mathcal{A}}|h_t}$ and its equivalent mean map vector $\hat{\mu}_{\mathcal{T}^{\mathcal{O}}\mathcal{T}^{\mathcal{A}}|h_t}$:

$$\begin{aligned}\widehat{C}_{\mathcal{T}^{\mathcal{O}},\mathcal{T}^{\mathcal{A}}|h_t} &= \Upsilon^{\mathcal{T}^{\mathcal{O}}}\Lambda_{h_t}\Upsilon^{\mathcal{T}^{\mathcal{A}}*} \\ &\equiv \hat{\mu}_{\mathcal{T}^{\mathcal{O}}\mathcal{T}^{\mathcal{A}}|h_t} = (\Upsilon^{\mathcal{T}^{\mathcal{O}}} \star \Upsilon^{\mathcal{T}^{\mathcal{A}}})\alpha_{h_t}\end{aligned} \tag{31}$$

where $\star$ is the Khatri-Rao (column-wise tensor) product. The last line is analogous to Eq. 6: each column of $\Upsilon^{\mathcal{T}^{\mathcal{O}}} \star \Upsilon^{\mathcal{T}^{\mathcal{A}}}$ is a single feature vector $\phi^{\mathcal{T}^{\mathcal{O}}}(\tau_t^{\mathcal{O}}) \otimes \phi^{\mathcal{T}^{\mathcal{A}}}(\tau_t^{\mathcal{A}})$ in the joint RKHS for test observations and test actions; multiplying by α_{h_t} gives a weighted average of these feature vectors.

Similarly, the HSE-PSR state can be written:

$$\begin{aligned}\widehat{\mathcal{S}}(h_t) &= \widehat{\mathcal{C}}_{\mathcal{T}^{\mathcal{O}},\mathcal{T}^{\mathcal{A}}|h_t}\widehat{\mathcal{C}}^{-1}_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}}|h_t} \\ &= \Upsilon^{\mathcal{T}^{\mathcal{O}}}(\Lambda_{h_t}G_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}}} + \lambda T I)^{-1}\Lambda_{h_t}\Upsilon^{\mathcal{T}^{\mathcal{A}}*} \\ &\equiv (\Upsilon^{\mathcal{T}^{\mathcal{O}}}(\Lambda_{h_t}G_{\mathcal{T}^{\mathcal{A}},\mathcal{T}^{\mathcal{A}}} + \lambda T I)^{-1} \star \Upsilon^{\mathcal{T}^{\mathcal{A}}})\alpha_{h_t}\end{aligned} \tag{32}$$

We can collect all the estimated HSE-PSR states, from all the histories in our training data, into one operator $\Upsilon^{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}}}$:

$$\mathcal{W}_{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}},h_{1:T}} \equiv \Upsilon^{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}}} = \left(\widehat{\mathcal{S}}(h_1), \ldots, \widehat{\mathcal{S}}(h_T)\right) \tag{33}$$

We need several similar operators which represent lists of vectorized conditional embedding operators. Write:

$$\Upsilon^{\mathcal{T}^{\mathcal{O}'}} = \left(\phi^{\mathcal{T}^{\mathcal{O}}}(\tau_2^{\mathcal{O}}), \ldots, \phi^{\mathcal{T}^{\mathcal{O}}}(\tau_{T+1}^{\mathcal{O}})\right) \tag{34}$$

$$\Upsilon^{\mathcal{T}^{\mathcal{A}'}} = \left(\phi^{\mathcal{T}^{\mathcal{A}}}(\tau_2^{\mathcal{A}}), \ldots, \phi^{\mathcal{T}^{\mathcal{A}}}(\tau_{T+1}^{\mathcal{A}})\right) \tag{35}$$

$$\Upsilon^{\mathcal{O}} = \left(\phi^{\mathcal{O}}(o_1), \ldots, \phi^{\mathcal{O}}(o_T)\right) \tag{36}$$

$$\Upsilon^{\mathcal{A}} = \left(\phi^{\mathcal{A}}(a_1), \ldots, \phi^{\mathcal{A}}(a_T)\right) \tag{37}$$

(Our convention is that primes indicate tests shifted forward in time by one step.) Now we can compute lists of: expected next HSE-PSR states $\mathcal{W}_{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'},h_{1:T}}$; embeddings of length-1 predictive distributions $\mathcal{W}_{\mathcal{O}|\mathcal{A},h_{1:T}}$; embeddings of length-1 predictive distributions $\mathcal{W}_{\mathcal{O},\mathcal{O}|\mathcal{A},h_{1:T}}$; and finally extended tests $\mathcal{W}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A},h_{1:T}}$. Vectorized, these become:

$$\mathcal{W}_{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'},h_{1:T}} = \Upsilon^{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'}} \tag{38}$$

$$\mathcal{W}_{\mathcal{O}|\mathcal{A},h_{1:T}} = \Upsilon^{\mathcal{O}|\mathcal{A}} \tag{39}$$

$$\mathcal{W}_{\mathcal{O},\mathcal{O}|\mathcal{A},h_{1:T}} = \Upsilon^{\mathcal{O},\mathcal{O}|\mathcal{A}} \tag{40}$$

$$\mathcal{W}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A},h_{1:T}} = \Upsilon^{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A}} \tag{41}$$

Each of these operators is computed analogously to Eqs. 32 and 33 above. The expanded columns of Eqs. 40 and 41 are of particular importance for future derivations:

$$\Upsilon_t^{\mathcal{O},\mathcal{O}|\mathcal{A}} = \Upsilon^{\mathcal{O},\mathcal{O}}(\Lambda_{h_t}G_{\mathcal{A},\mathcal{A}} + \lambda T I)^{-1}\Lambda_{h_t}\Upsilon^{\mathcal{A}*} \tag{42}$$

$$\Upsilon_t^{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A}} = \Upsilon^{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'}}(\Lambda_{h_t}G_{\mathcal{A},\mathcal{A}} + \lambda T I)^{-1}\Lambda_{h_t}\Upsilon^{\mathcal{A}*} \tag{43}$$

Finally, the finite-dimensional product of any two lists of vectorized states is a Gram matrix. In particular, we need $G_{\mathcal{T},\mathcal{T}}$ and $G_{\mathcal{T},\mathcal{T}'}$, Gram matrices corresponding to HSE-PSR states and time-shifted HSE-PSR states:

$$G_{\mathcal{T},\mathcal{T}} = \Upsilon^{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}}*}\Upsilon^{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}}} \tag{44}$$

$$G_{\mathcal{T},\mathcal{T}'} = \Upsilon^{\mathcal{T}^{\mathcal{O}}|\mathcal{T}^{\mathcal{A}}*}\Upsilon^{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'}} \tag{45}$$

5.2 LEARNING THE UPDATE RULE

The above derivation shows how to get a state estimate by embedding an entire history; for a dynamical system model, though, we want to avoid remembering the entire history, and instead recursively update the state of the HSE-PSR given new actions and observations. We are now in a position to do so. We first show how to learn a feasible HSE-PSR state that we can use to initialize filtering (Section 5.2.1), and then show how to learn the prediction operators (Section 5.2.2). Finally, we show how to perform filtering with KBR (Section 5.3).

5.2.1 Estimating a Feasible State

If our data consists of a single long trajectory, we *cannot* estimate the initial state $\mathcal{S}_0$, since we only see the null history once. So, instead, we will estimate an arbitrary feasible state $\mathcal{S}_*$, which is enough information to enable prediction after an initial tracking phase if we assume that our process mixes. If we have multiple trajectories, a straightforward modification of (46) will allow us to estimate $\mathcal{S}_0$ as well.

In particular, we take $\mathcal{S}_*$ to be the RKHS representation of the stationary distribution of core test predictions given the blind policy that we used to collect the data. We estimate $\mathcal{S}_*$ as the empirical average of state estimates: $\widehat{\mathcal{W}}_{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A},h_*} = \Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}}\alpha_{h_*}$ where

$$\alpha_{h_*} = \frac{1}{T}\mathbf{1}_T \tag{46}$$

5.2.2 Estimating the Prediction Operators

The linear prediction operators $\mathcal{F}_{\mathcal{AO}}$ and $\mathcal{F}_{\mathcal{AOT}}$ from Eqs. 18 and 19 are the critical parameters of the HSE-PSR used to update state. In particular, we note that $\mathcal{F}_{\mathcal{AO}}$ is a linear mapping from $\mathcal{W}_{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A},h_t}$ to $\mathcal{W}_{\mathcal{O}|\mathcal{A},h_t}$ and $\mathcal{F}_{\mathcal{AOT}}$ is a linear mapping from $\mathcal{W}_{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A},h_t}$ to $\mathcal{W}_{\mathcal{T}^\mathcal{O},\mathcal{O}|\mathcal{T}^\mathcal{A},\mathcal{A},h_t}$. So, we estimate these prediction operators by kernel ridge regression:

$$\widehat{\mathcal{F}}_{\mathcal{AO}} = \Upsilon^{\mathcal{O},\mathcal{O}|\mathcal{A}}\left(\Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}}\right)^\dagger_{\lambda T} \tag{47}$$

$$\widehat{\mathcal{F}}_{\mathcal{AOT}} = \Upsilon^{\mathcal{T}^\mathcal{O},\mathcal{O}|\mathcal{T}^\mathcal{A},\mathcal{A}}\left(\Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}}\right)^\dagger_{\lambda T} \tag{48}$$

These operators are (possibly) infinite-dimensional, so we never actually build them; instead, we show how to use Gram matrices to apply these operators implicitly.

5.3 GRAM MATRIX STATE UPDATES

We now apply kernel Bayes' rule to *update* state given a new action and observation, i.e., to implement Eqs. 18–22 via Gram matrices. We start from the current weight vector α_t, which represents our current state $\mathcal{S}(h_t) = \Upsilon^{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'}}\alpha_t$.

Predicting forward in time means applying Eqs. 47 and 48 to state. We do this in several steps. First we apply the regularized pseudoinverse in Eqs. 47 and 48, which can be written in terms of Gram matrices:

$$\begin{aligned}\left(\Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}}\right)^\dagger_{\lambda T} &= \Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}*}\left(\Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}}\Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}*} + \lambda TI\right)^{-1}\\ &= (G_{\mathcal{T},\mathcal{T}}+\lambda TI)^{-1}\Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}*}\end{aligned} \tag{49}$$

Applying Eq. 49 to the state $\Upsilon^{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'}}\alpha_t$ results in

$$\begin{aligned}\hat{\alpha}_t &= (G_{\mathcal{T},\mathcal{T}} + \lambda TI)^{-1}\Upsilon^{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}*}\Upsilon^{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'}}\alpha_t\\ &= (G_{\mathcal{T},\mathcal{T}} + \lambda TI)^{-1}G_{\mathcal{T},\mathcal{T}'}\alpha_t\end{aligned} \tag{50}$$

Here the weight vector $\hat{\alpha}_t$ allows us to predict the extended tests at time t conditioned on actions and observations up to time $t-1$. That is, from Eqs. 47, 48 and 50 we can write the estimates of Eqs. 18 and 19:

$$\mathcal{F}_{\mathcal{AO}}\mathcal{S}(h_t) = \Upsilon^{\mathcal{O},\mathcal{O}|\mathcal{A}}\hat{\alpha}_t$$
$$\mathcal{F}_{\mathcal{AOT}}\mathcal{S}(h_t) = \Upsilon^{\mathcal{T}^\mathcal{O},\mathcal{O}|\mathcal{T}^\mathcal{A},\mathcal{A}}\hat{\alpha}_t$$

And, from Eqs. 42 and 43 we see that

$$\Upsilon^{\mathcal{O},\mathcal{O}|\mathcal{A}}\hat{\alpha}_t = \sum_{i=1}^T[\hat{\alpha}_t]_i\,\Upsilon^{\mathcal{O},\mathcal{O}}(\Lambda_{h_i}G_{\mathcal{A},\mathcal{A}} + \lambda TI)^{-1}\Lambda_{h_i}\Upsilon^{\mathcal{A}*} \tag{51}$$

$$\Upsilon^{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},\mathcal{A}}\hat{\alpha}_t = \sum_{i=1}^T[\hat{\alpha}_t]_i\,\Upsilon^{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'}}(\Lambda_{h_i}G_{\mathcal{A},\mathcal{A}} + \lambda TI)^{-1}\Lambda_{h_i}\Upsilon^{\mathcal{A}*} \tag{52}$$

After choosing action a_t, we can condition the embedded tests by right-multiplying Eqs. 51 and 52 by $\phi^\mathcal{A}(a_t)$. We do this by first collecting the common part of Eqs. 51 and 52 into a new weight vector α_t^a:

$$\alpha_t^a = \sum_{i=1}^T[\hat{\alpha}_t]_i\,(\Lambda_{h_i}G_{\mathcal{A},\mathcal{A}}+\lambda TI)^{-1}\Lambda_{h_i}\Upsilon^{\mathcal{A}*}\phi^\mathcal{A}(a_t) \tag{53}$$

The estimated conditional embeddings (Eqs. 20–21) are therefore:

$$\widehat{\mathcal{C}}_{\mathcal{O},\mathcal{O}|h_t,a_t} = \Upsilon^{\mathcal{O},\mathcal{O}}\alpha_t^a$$
$$\widehat{\mathcal{W}}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},h_t,a_t} = \Upsilon^{\mathcal{T}^{\mathcal{O}'}\mathcal{O}|\mathcal{T}^{\mathcal{A}'}}\alpha_t^a$$

Or, given a diagonal matrix with the weights α_t^a along the diagonal, $\Lambda_t^a = \mathrm{diag}(\alpha_t^a)$, the estimated conditional embeddings can be written:

$$\widehat{\mathcal{C}}_{\mathcal{O},\mathcal{O}|h_t,a_t} = \Upsilon^\mathcal{O}\Lambda_t^a\Upsilon^{\mathcal{O}*} \tag{54}$$

$$\widehat{\mathcal{W}}_{\mathcal{T}^{\mathcal{O}'},\mathcal{O}|\mathcal{T}^{\mathcal{A}'},h_t,a_t} = \Upsilon^{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'}}\Lambda_t^a\Upsilon^{\mathcal{O}*} \tag{55}$$

Given a new observation o_t, we apply KBR (Eq. 22):

$$\alpha_t^{ao} = (\Lambda_t^aG_{\mathcal{O},\mathcal{O}} + \lambda TI)^{-1}\Lambda_t^a\Upsilon^{\mathcal{O}*}\phi^\mathcal{O}(o_t) \tag{56}$$

Finally, given the coefficients α_t^{ao}, the HSE-PSR state at time $t+1$ is:

$$\widehat{\mathcal{S}}(h_t) = \widehat{\mathcal{W}}_{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'},h_{t+1}} = \Upsilon^{\mathcal{T}^{\mathcal{O}'}|\mathcal{T}^{\mathcal{A}'}}\alpha_t^{ao} \tag{57}$$

This completes the state update. The nonparametric state at time $t+1$ is represented by the weight vector $\alpha_{t+1} = \alpha_t^{ao}$. We can continue to recursively filter on actions and observations by repeating Eqs. 50–57.

6 PREDICTIONS

In the previous sections we have shown how to maintain the HSE-PSR state by implicitly tracking the operator $\mathcal{W}_{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A}}$. However, the goal of modeling a stochastic process is usually to make *predictions*, i.e., reason about properties of future observations. We can do so via mean embeddings: for example, given the state after some history h, $\mathcal{W}_{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A},h}$, we can fill in a sequence of test actions to find the mean embedding of the distribution over test observations:

$$\mu_{\mathcal{T}^\mathcal{O}|h,a_{1:M}} = \mathcal{W}_{\mathcal{T}^\mathcal{O}|\mathcal{T}^\mathcal{A},h}\phi^{\mathcal{T}^\mathcal{A}}(a_{1:M}) \tag{58}$$

As is typical with mean embeddings, we can now predict the expected value of any function f in our RKHS:

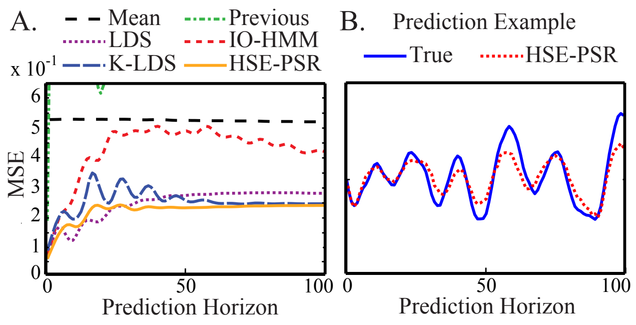


Figure 1: Synthetic data prediction performance. (A) Mean Squared Error for prediction with different estimated models. Each model was evaluated 1000 times; see text for details. (B) Example of the HSE-HMM's predicted observations given a sequence of 100 control inputs. As expected, the prediction is very accurate at the current time-step but degrades over time.

$$\mathbb{E}[f(o_{1:M}) \mid h, a_{1:M}] = \langle f, \mu_{\mathcal{T}^{\mathcal{O}}|h,a_{1:M}} \rangle \tag{59}$$

The range of predictions we can make therefore depends on our RKHS. For example, write $\pi_{ij}(o_{1:M})$ for the function which extracts the ith coordinate of the jth future observation. If these coordinate projections are in our RKHS, we can compute $\mathbb{E}[(o_j)_i \mid h, a_{1:M}]$ as the inner product of $\mu_{\mathcal{T}^{\mathcal{O}}|h,a_{1:M}}$ with π_{ij}. (Coordinate projection functions are present, for example, in the RKHS for a polynomial kernel, or in the RKHS for a Gaussian kernel on any compact subset of a real vector space.) Or, if our RKHS contains an indicator function for a region A, we can predict the probability that the future observations fall in A.

Sometimes the desired function is absent from our RKHS. In this case, we can learn an approximation from our training data by kernel linear regression. This approximation has a particularly simple and pleasing form: we compute $f_s = f(o_{s:s+M-1})$ at each training time point s, collect these f_s into a single vector f, and predict $\mathbb{E}[f(o_{1:M}) \mid h, a_{1:M}] = f^\top \alpha_h$, where α_h is the vector of weights representing our state after history h. In the experiments in Section 7 below, we use this trick to evaluate the expected next observation.

7 EXPERIMENTS

7.1 SYNTHETIC DATA

First we tested our algorithm on a benchmark synthetic nonlinear dynamical system [21, 22]:

$$\begin{aligned}\dot{x}_1(t) =& \; x_2(t) - 0.1\cos(x_1(t))\left(5x_1(t) - 4x_1^3(t) + x_1^5(t)\right)\\ &- 0.5\cos(x_1(t))\,u(t),\\ \dot{x}_2(t) =& -65x_1(t) + 50x_1^3(t) - 15x_1^5(t) - x_2(t) - 100u(t),\\ y(t) =& \; x_1(t)\end{aligned}$$

The output is y; the policy for the control input u is zero-order hold white noise, uniformly distributed between -0.5 and 0.5. We collected a single trajectory of 1600 observations and actions at 20Hz, and split it into 500 training and 1200 test data points.

For each model, discussed below, we filtered for 1000 different extents $t_1 = 101, \ldots, 1100$, then predicted the system output a further t_2 steps in the future, for $t_2 = 1, \ldots, 100$. We averaged the squared prediction error over all t_1; results are plotted in Figure 1(A).

We trained a HSE-PSR using the algorithm described in Section 5 with Gaussian RBF kernels and tests and histories consisting of 10 consecutive actions and observations. The bandwidth parameter of the Gaussian RBF kernels is set with the "median trick." For comparison, we learned several additional models with parameters set to maximize each model's performance: a 5-dimensional nonlinear model using a kernelized version of linear system identification (K-LDS) [22], a 5-dimensional linear dynamical system (LDS) using a stabilized version of spectral subspace identification [23, 24] with Hankel matrices of 10 time steps; and a 50-state input-output HMM (IO-HMM) trained via EM [2], with observations and actions discretized into 100 bins. We also compared to simple baselines: the mean observation and the previous observation. The results (Figure 1(A)) demonstrate that the HSE-PSR algorithm meets or exceeds the performance of the competing models.

7.2 SLOT CAR

The second experiment was to model inertial measurements from a slot car (1:32 scale) racing around a track. Figure 2(A) shows the car and attached 6-axis IMU (an Intel Inertiadot), as well as the 14m track. (Song et al. [20, 10] used a similar dataset.) We collected the estimated 3D acceleration and angles of the car (observations) from the IMU as well as the velocity of the car (the control input) at 10Hz for 2500 steps. We split our data into 500 training and 2000 test data points. The control policy was designed to maximize speed—it is not blind, but our learning algorithm works well despite this fact.

For each model, we performed filtering for 1000 different extents $t_1 = 501, \ldots, 1500$, then predicted an IMU reading a further t_2 steps in the future, for $t_2 = 1, \ldots, 500$, using the given control signal. We averaged the squared prediction error over all t_1; results are plotted in Figure 2(B).

The models are: an HSE-PSR with Gaussian RBF kernels on tests and histories consisting of 150 consecutive actions and observations; a 40-dimensional nonlinear model trained by K-LDS with the same settings as our HSE-PSR; a stablized 40-dimensional LDS with Hankel matrices of 150 time steps; and a 50-state IO-HMM, with observations and actions discretized into

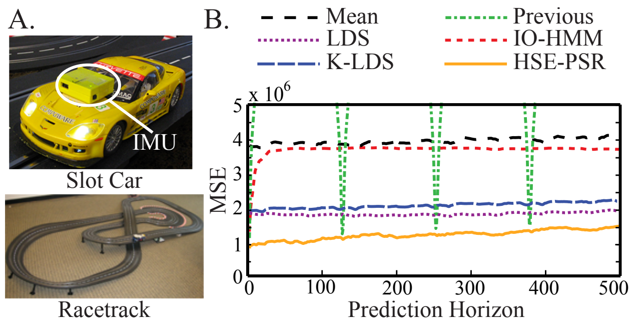


Figure 2: Slot car experiment. (A) The slot car platform: the car and IMU (top) and the racetrack (bottom). (B) Mean Squared Error for prediction with different estimated models. Each model was evaluated 1000 times; see text for details.

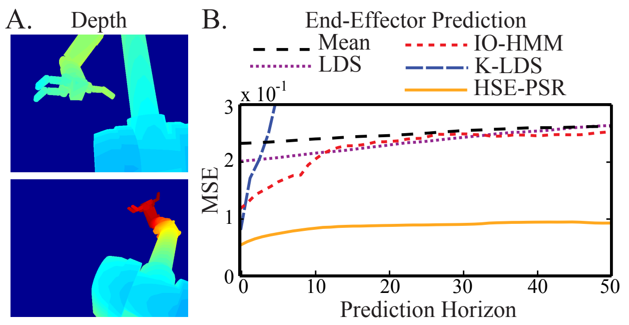


Figure 3: Robot end-effector prediction. (A) Observations consisted 640x480 pixel depth images of a robot arm. (B) Mean Squared Error (in cm) for end-effector prediction with different learned models. Each model was evaluated 400 times; see text for details.

200 bins. We again included mean and previous observation as baselines.[4] In general, the dynamical systems designed for continuous observations and controls performed well, but the HSE-PSR consistently yields the lowest RMSE.

7.3 ARM END-EFFECTOR PREDICTION

In the final experiment we look at the problem of predicting the 3-d position of the end-effector of a simulated Barrett WAM robot arm observed by a depth-camera. Figure 3(A) shows example depth images.

We collected 1000 successive observations of the arm motor babbling. The data set consisted of depth maps and the 3D position of the end-effector along with the joint angles of the robot arm (which we treat as the control signal). The goal was to learn a nonlinear dynamical model of the depth images and 3D locations in response to the joint angles, both to remove noise and to account for hysteresis in the reported angles. After filtering on the joint angles and depth images, we predict current and future 3D locations of the end-effector. We used the first 500 data points as training data, and held out the last 500 data points for testing the learned models.

For each model described below, we performed filtering for 400 different extents $t_1 = 51, \ldots, 450$ based on the depth camera data and the joint angles, then predicted the end effector position a further t_2 steps in the future, for $t_2 = 1, 2..., 50$ using just the inputs. The squared error of the predicted end-effector position was recorded, and averaged over all of the extents t_1 to obtain the means plotted in Figure 2(B).

We trained a HSE-PSR with Gaussian RBF kernels and tests and histories consisting of 5 consecutive actions and observations. For comparison, we learned a 100-dimensional nonlinear model using K-LDS with the same settings as our HSE-PSR, a stabilized 100-dimensional LDS with Hankel matrices of 5 time steps; and a 100-state discrete IO-HMM where observations and actions were discretized into 100 values. This is a very challenging problem and most of the approaches had difficulty making good predictions. For example, the K-LDS learning algorithm generated an unstable model and the stabilized LDS had poor predictive accuracy. The HSE-PSR yields significantly lower mean prediction error compared to the alternatives.

8 CONCLUSION

In this paper we attack the problem of learning a controlled stochastic process directly from sequences of actions and observations. We propose a novel and highly expressive model: *Hilbert space embeddings of predictive state representations*. This model extends discrete linear PSRs to large and continuous-valued dynamical systems. With the proper choice of kernel, HSE-PSRs can represent near-arbitrary continuous and discrete-valued stochastic processes.

HSE-PSRs also admit a powerful learning algorithm. As with ordinary PSRs, the parameters of the model can be written entirely in terms of predictive distributions of observable events. (This is in stark contrast to latent variable models, which have unobservable parameters that are usually estimated by heuristics such as EM.) Unlike previous work on continuous-valued PSRs, we do not assume that predictive distributions conform to particular parametric families. Instead, we define the HSE-PSR state as the nonparametric embedding of a conditional probability operator in a characteristic RKHS, and use recent theory developed for RKHS embeddings of distributions to derive sample-based learning and filtering algorithms.

[4]Like a stopped clock, the previous observation (the green dotted line) is a good predictor every 130 steps or so as the car returns to a similar configuration on the track.

References

[1] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE*, 77(2):257–285, 1989.

[2] Yoshua Bengio and Paolo Frasconi. An Input Output HMM Architecture. In *Advances in Neural Information Processing Systems*, 1995.

[3] Michael Littman, Richard Sutton, and Satinder Singh. Predictive representations of state. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

[4] Byron Boots, Sajid M. Siddiqi, and Geoffrey J. Gordon. Closing the learning-planning loop with predictive state representations. In *Proceedings of Robotics: Science and Systems VI*, 2010.

[5] David Wingate and Satinder Singh. Exponential family predictive representations of state. In *Proc. NIPS*, 2007.

[6] David Wingate and Satinder Singh. On discovery and learning of models with predictive representations of state for agents with continuous actions and observations. In *Proc. AAMAS*, 2007.

[7] A.J. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In E. Takimoto, editor, *Algorithmic Learning Theory*, Lecture Notes on Computer Science. Springer, 2007.

[8] B. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. Injective Hilbert space embeddings of probability measures. 2008.

[9] Kenji Fukumizu, Le Song, and Arthur Gretton. Kernel bayes' rule. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1737–1745. 2011.

[10] L. Song, B. Boots, S. M. Siddiqi, G. J. Gordon, and A. J. Smola. Hilbert space embeddings of hidden Markov models. In *Proc. 27th Intl. Conf. on Machine Learning (ICML)*, 2010.

[11] Byron Boots and Geoffrey Gordon. Two-manifold problems with applications to nonlinear system identification. In *Proc. 29th Intl. Conf. on Machine Learning (ICML)*, 2012.

[12] Steffen Grunewalder, Guy Lever, Luca Baldassarre, Massimiliano Pontil, and Arthur Gretton. Modelling transition dynamics in MDPs with RKHS embeddings. *CoRR*, abs/1206.4655, 2012.

[13] Y Nishiyama, A Boularias, A Gretton, and K Fukumizu. Hilbert space embeddings of POMDPs. 2012.

[14] Judea Pearl. *Causality: models, reasoning, and inference*. Cambridge University Press, 2000.

[15] Michael Bowling, Peter McCracken, Michael James, James Neufeld, and Dana Wilkinson. Learning predictive state representations using non-blind policies. In *Proc. ICML*, 2006.

[16] Satinder Singh, Michael James, and Matthew Rudary. Predictive state representations: A new theory for modeling dynamical systems. In *Proc. UAI*, 2004.

[17] C. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

[18] L. Song, J. Huang, A. Smola, and K. Fukumizu. Hilbert space embeddings of conditional distributions. In *International Conference on Machine Learning*, 2009.

[19] S. Grunewalder, G. Lever, L. Baldassarre, S. Patterson, A. Gretton, and M. Pontil. Conditional mean embeddings as regressors. In *ICML*, 2012.

[20] Byron Boots and Geoffrey Gordon. An online spectral learning algorithm for partially observable nonlinear dynamical systems. In *Proceedings of the 25th National Conference on Artificial Intelligence (AAAI-2011)*, 2011.

[21] V. Verdult, J.A.K. Suykens, J. Boets, I. Goethals, and B. De Moor. Least squares support vector machines for kernel cca in nonlinear state-space identification. In *Proceedings of the 16th international symposium on Mathematical Theory of Networks and Systems (MTNS2004), Leuven, Belgium*, 2004.

[22] Yoshinobu Kawahara, Takehisa Yairi, and Kazuo Machida. A kernel subspace method by stochastic realization for learning nonlinear dynamical systems, 2006.

[23] B. Boots. Learning Stable Linear Dynamical Systems. Data Analysis Project, Carnegie Mellon University, 2009.

[24] Sajid Siddiqi, Byron Boots, and Geoffrey J. Gordon. A constraint generation approach to learning stable linear dynamical systems. In *Proc. NIPS*, 2007.

Scoring and Searching over Bayesian Networks with Causal and Associative Priors

Giorgos Borboudakis
Comp. Sci. Dept., University of Crete
Institute of Computer Science, FORTH

Ioannis Tsamardinos
Comp. Sci. Dept., University of Crete
Institute of Computer Science, FORTH

Abstract

A significant theoretical advantage of search-and-score methods for learning Bayesian Networks is that they can accept informative prior beliefs for each possible network, thus complementing the data. In this paper, a method is presented for assigning priors based on beliefs on the presence or absence of certain paths in the true network. Such beliefs correspond to knowledge about the possible causal and associative relations between pairs of variables. This type of knowledge naturally arises from prior experimental and observational data, among others. In addition, a novel search-operator is proposed to take advantage of such prior knowledge. Experiments show that, using path beliefs improves the learning of the skeleton, as well as the edge directions in the network.

1 INTRODUCTION

One theoretical advantage of the search-and-score approach to learning Bayesian Networks [Cooper and Herskovits, 1992] versus the constraint-based approach [Spirtes et al., 2000] is that the former naturally accepts priors for each network. Since the number of possible networks is exponential to the number of nodes, in a practical setting one has to assign priors in an implicit way. In this paper, we consider prior beliefs on the possible paths between variable pairs. Such paths directly correspond to **causal** or **associative** relations. The joint beliefs on the paths is then employed to assign a prior on each network.

Causal knowledge naturally derives from prior experimental data while associative knowledge stems from observational data. For example, consider a dataset $\mathcal{D}$ measuring the average amount of exercise per week E, calcium in diet C, occurrence of osteoporosis by 60yrs O and smoking S in a cohort of women. A Bayesian Network could be induced by any appropriate learning method. However, if a prior *experimental study* showed that increasing the amount of exercise reduces the occurrence of the disease, then the knowledge belief that [E causes (that is, causally affects) O] with probability p should be incorporated during learning. Similarly, if a prior cohort study (observational study) has shown that smoking correlates with reduced exercising, then knowledge [S and E are associated] with probability p' should also be included. The belief strengths p and p' depend on several factors, such as the statistical power of the study. Notice that the fact [E causes O] *does not* correspond to the presence of the edge $E \rightarrow O$ in the network: the edge implies a *direct* causal relation (in some context of modeled variables) while [E causes O] does not depend on the context.

Path beliefs are inherently dependent. For example, if one believes with certainty that [X causes Y] and [Y causes Z], then one has to believe that [X causes Z] to be consistent. Therefore, one should consider the joint distribution of the input path beliefs, instead of the marginal distributions separately. However, it is very unlikely that the complete joint distribution is available. Instead, one could use the marginal distributions to infer a joint distribution. However, there are several technical difficulties to consider. For example, assume we are given $P(X \text{ causes } Y) = 0.8$ and $P(Y \text{ causes } Z) = 0.8$ and wish to compute $P(X \text{ causes } Y, Y \text{ causes } Z)$. On one hand, there may be several choices for the joint given the same marginal beliefs. In the above scenario we can infer $P(X \text{ causes } Y, Y \text{ causes } Z) \in [0.6, 1]$. On the other hand, the beliefs maybe *incoherent* [Hansen et al., 2000], that is, not extendable to a joint distribution that satisfies the probability axioms.

We present a method that computes a joint distribution of the path beliefs such that: if the path beliefs are coherent the joint is the closest to uninformative priors; if they are incoherent the joint is chosen to be coherent and induces path probabilities that are close to the input beliefs. Once the joint is computed, it can be employed to efficiently compute the prior of a network. Furthermore, to take advantage of the prior knowledge, we introduce a novel search-operator.

In simulated proof-of-concept experiments we show that the new scoring method can indeed take advantage of prior knowledge. When provided with causal knowledge, it is able to better learn *the orientations* of the edges and the causal relations. Informative priors can also facilitate learning *the skeleton* of the network. Finally, we show that the proposed search-operator significantly improves the quality of the learned model.

There are several other methods that make use of prior knowledge when learning a network (see [Angelopoulos and Cussens, 2008] for a review). For example, using knowledge regarding the parameters of the network [Niculescu et al., 2006], a causal total order of the variables [Cooper and Herskovits, 1992], the presence or absence of directed edges in the network [Meek, 1995] possibly with beliefs assigned to them [Buntine, 1991, Robert and Arno, 2000], or a prior network, used to assign prior probabilities to each network based on the distance from this network [Heckerman et al., 1995]. In general, it can be argued that the type of knowledge the existing methods can incorporate during learning is not in a form that can be easily acquired. As a result, uniform - and thus uninformative - priors are commonly used when learning Bayesian Networks from data. *The problem of incorporating informative priors while learning is listed in the list of open problems in a recent causality editorial* [Spirtes, 2010].

There also is prior work that specifically considers path constraints or beliefs. The methods in [Borboudakis et al., 2011, Borboudakis and Tsamardinos, 2012] assume one *first learns* a Markov-Equivalence class of Bayesian Networks or Maximal Ancestral Graphs [Spirtes et al., 2000] (a generalization of Bayesian Networks that admits hidden variables) from data and *then*, path constraints are imposed on the graph. In contrast, in this work the network is learned *with the help of the prior knowledge.* In [O'Donnell et al., 2006] a method is presented for incorporating beliefs on paths, but relies on computationally expensive Markov Chain Monte Carlo (MCMC) simulations. However, neither the latter, nor any other method dealing with prior knowledge deals with the issues of dependent and possibly incoherent beliefs.

2 BACKGROUND

We assume the reader's familiarity with Bayesian Networks [Pearl, 2000, Neapolitan, 2003] and learning algorithms and just briefly review the basic concepts.

Let $\mathcal{V}$ be a set of k random variables $\{V_i\}_{i=1}^k$. A **Bayesian Network** (BN) over $\mathcal{V}$ is a pair $\mathcal{B} = \langle \mathcal{G}_\mathcal{V}, \mathcal{P}_\mathcal{V} \rangle$, where $G_\mathcal{V}$ is a **Directed Acyclic Graph** (DAG) representing conditional independencies between variables $\mathcal{V}$, and $P_\mathcal{V}$ is the joint probability distribution (j.p.d.) of $\mathcal{V}$. The graph and distribution must be connected by the equation $P_\mathcal{V} = \prod P(V_i|Pa_\mathcal{G}(V_i))$, where $Pa_\mathcal{G}(V_i)$ are the parents of V_i in $\mathcal{G}$. The above equation is equivalent to what is called the **Markov Condition**. When the network is fixed in a context we drop the indexes $\mathcal{V}, \mathcal{G}$ from the equations. The **skeleton** of a BN $\mathcal{G}$ is the undirected graph which can be constructed by ignoring the orientations of $\mathcal{G}$. A triple of vertices $\langle X, Y, Z \rangle$ is called a **collider** in $\mathcal{G}$, if $X \rightarrow Y \leftarrow Z$ is in $\mathcal{G}$. A collider $\langle X, Y, Z \rangle$ is **unshielded** if X and Z are not adjacent in $\mathcal{G}$. Two BNs are called **Markov equivalent** if: (a) they have the same skeleton, and (b) they have the same set of unshielded colliders. A **Partially Directed Acyclic Graph** (PDAG) (also known as essential graph) is a graph representing a set of Markov equivalent BNs. It has the same skeleton as all BN representatives and an edge is directed if and only if it is invariant in all BN representatives. A **directed path** from X to Y is denoted as $X \Rightarrow Y$. We **denote as** $X \Leftrightarrow Y$ the case where X and Y share a common ancestor in $\mathcal{G}$, but neither X is an ancestor of Y nor the reverse. A *d*-**connecting path** (given the empty set) between X and Y exists if either $X \Rightarrow Y$, $X \Leftarrow Y$, or $X \Leftrightarrow Y$. The absence of a *d*-connecting path between X and Y is **denoted as** $X \not\Leftrightarrow Y$. In the rest of the paper, we assume the Faithfulness Condition [Spirtes et al., 2000] that (together with the Markov Condition) implies that *there is a d-connecting path between X and Y, if and only if the two nodes are statistically associated (dependent).*

Let $\mathcal{D}$ be a complete multinomial dataset over variables $\mathcal{V}$. The probability of a network $\mathcal{G}$ over $\mathcal{V}$ is $P(G|D) \propto P(D|G) \cdot P(G)$. The score of a network is often obtained by taking the logarithm of $P(G|D)$, and equals $Sc(G|D) = Sc(D|G) + Sc(G)$. Bayesian scoring methods such as K2 [Cooper and Herskovits, 1992] and BDe, BDeu, [Heckerman et al., 1995] try to approximate the log-likelihood based on different assumptions. When priors are uniform, the term $Sc(G)$ can be ignored during maximization. In our setting however, this term may become important.

3 REPRESENTING PATH BELIEFS

For any pair $X, Y \in \mathcal{V}$ we may have a prior belief on the possible paths connecting the two nodes in the network. It is important that we devise cases for such paths that are *mutually exclusive* and *allow the representation of common types of causal and associa-*

tive knowledge. This is possible as follows: we define the **path variables** $r_{i,j}$ taking values in the domain $\{\Rightarrow, \Leftarrow, \Leftrightarrow, \not\Leftrightarrow\}$ with the semantics $V_i \Rightarrow V_j$, $V_i \Leftarrow V_j$, $V_i \Leftrightarrow V_j$, and $V_i \not\Leftrightarrow V_j$ respectively. When the specific nodes V_i, V_j we refer to are not important we will use a single index: r_k. Each variable $r_{i,j}$ has a probability distribution $\Pi_{r_{i,j}} = \langle \pi_{\Rightarrow}, \pi_{\Leftarrow}, \pi_{\Leftrightarrow}, \pi_{\not\Leftrightarrow} \rangle$ over each possible value. The input to our method is a set of path beliefs $\mathbf{K} = \langle \mathbf{R}, \mathbf{\Pi} \rangle$, where $\mathbf{R}$ is a set of path variables and $\mathbf{\Pi}$ the set of probability distributions associated with them. An example is shown in Table 1a(Top) expressing the belief that most likely there is a directed path from X to Y and from Y to Z.

The possible paths between nodes dictate their possible **causal** and **associative** relations. When the BN is interpreted causally, then $X \Rightarrow Y$ is equivalent to [X causes Y]. In addition, as discussed in the previous section: $X \Rightarrow Y$ or $X \Leftarrow Y$ or $X \Leftrightarrow Y$ is equivalent to [X is associated with Y]. Thus, a distribution $\Pi_{r_{X,Y}} = \langle \pi_{\Rightarrow}, \pi_{\Leftarrow}, \pi_{\Leftrightarrow}, \pi_{\not\Leftrightarrow} \rangle$ corresponds to beliefs about the causal and associative relations.

In practice, it is useful to allow the user to specify prior beliefs directly on the events [X does (not) cause Y] and [X is (not) associated with Y] from which the distribution $\Pi_{r_{XY}}$ can be derived, than the opposite. This is not difficult: for example given $P(X \text{ causes } Y) = \pi_{\Rightarrow}$ the mass of probability $1 - \pi_{\Rightarrow}$ has to be distributed in a reasonable way to the other three values. However, we avoid this belief representation to simplify the presentation of the method.

4 SCORING PATH BELIEFS

In this section, we derive a score for DAG G given data D and n path beliefs in $\mathbf{K}$. An important requirement for the computation of the score is knowledge of a joint distribution $J = P(r_1, \ldots, r_n|\mathbf{\Pi}) = P(\mathbf{R}|\mathbf{\Pi})$ such that its marginals correspond to the distributions in $\mathbf{\Pi}$. We assume J is already computed; the following sections describe the details of this computation. The j.p.d. J stemming from $\mathbf{K}$ in Table 1a is shown in Table 1b.

We denote with C (configuration) a given joint instantiation of values to path variables $\mathbf{R} = \langle r_1, \ldots r_n \rangle$, and define $J_C = P(\mathbf{R} = C|\mathbf{\Pi})$. It is important to notice that *for each graph G the configuration C is uniquely determined*. For example, in the j.p.d. of Table 1b, if in a graph G $X \Rightarrow Y$, $Y \Rightarrow Z$ and $X \Rightarrow Z$ hold, then $\mathbf{r} = C_1$. Thus, it makes sense to denote with C_G the joint instantiation of variables $\mathbf{R}$ in graph G.

Let G be a DAG and D a dataset over the same variables. We now compute the probability $P(G|D, J)$:

$$P(G|D,J) = \frac{P(D|G,J) \cdot P(G|J)}{P(D|J)} = \frac{P(D|G) \cdot P(G|J)}{P(D|J)}$$

The second equation stems from the fact that given the graph G the data D are independent of J (J does not provide any additional information about the data once the graph is known). The factor $P(D|J)$ is a normalizing constant that does not need be computed when we maximize the above equation over different graphs. In Section 2 we mention several approximations for computing the factor $P(D|G)$. We now focus on the prior $P(G|J)$:

$$\begin{aligned} P(G|J) =& P(G, C_G|J) = P(G|J, C_G) \cdot P(C_G|J) = \\ & P(G|C_G) \cdot P(C_G|J) = P(G|C_G) \cdot J_{C_G} \end{aligned}$$

The first equation holds because C_G is a function of G. The factor $P(G|C_G)$ is our prior on a graph G given that a specific configuration holds. Given no other preference or knowledge *we assign the same prior to all graphs with the same configuration*. Let N_C be the number of DAGs over nodes $\mathcal{V}$ sharing the same configuration C. Then $P(G|C_G) = 1/N_{C_G}$ and so:

$$P(G|J) = \frac{J_{C_G}}{N_{C_G}} \quad \text{and} \quad Sc(G|J) = \log\left(\frac{J_{C_G}}{N_{C_G}}\right) \quad (1)$$

The overall score of a graph is then defined as:

$$Sc(G|D,J) = Sc(D|G) + Sc(G|J) \quad (2)$$

The score $Sc(G|D, J)$ has two desirable properties:

1. **Markov-Equivalent graphs that satisfy the same path-beliefs obtain the same score.** The last term in the equation above is the same for graphs sharing the same configuration. The first term is the same for Markov-equivalent graphs provided one employs an appropriate scoring function, such as the BDe score [Heckerman et al., 1995].
2. **For uninformative prior beliefs, all graphs are equiprobable a priori**, that is, $P(G|J) = 1/N$, where N is the number of graphs over nodes $\mathcal{V}$. With uninformative beliefs we expect to encounter a given configuration with probability equal to the proportion of the graphs satisfying the configuration, i.e,. $J_C = \frac{N_C}{N}$. In that case, $P(G|J) = \frac{N_C}{N} \cdot \frac{1}{N_C} = \frac{1}{N}$ and we end up with uniform priors as we would expect.

While Eq. 1 follows the above two properties, we point out to the fact that the factor $1/N_{C_G}$ may seem to provide counter-intuitive results at a first glance. The reason is that, everything else being equal, higher priors will tend to be assigned to graphs in "small" configurations, that is, consistent with only a few graphs. If this is not desirable then one can drop the $1/N_C$ factor. However, if this score is used in place of Eq. 1 then Property 2 above is not satisfied any more.

Table 1: (a) (Top Part) Path beliefs $\mathbf{K}$ for three pairs of nodes. The beliefs are *incoherent*: $P(X \Rightarrow Y) = 0.8$ and $P(Y \Rightarrow Z) = 0.9$ imply that $P(X \Rightarrow Z) \in [0.7, 1]$. (a) (Bottom Part) Induced *coherent* beliefs $\mathbf{K}'$ stemming from $\mathbf{K}$ by solving the problem in Eq. 6. (b) A part of the j.p.d. J computed by solving Eq. 6 with input $\mathbf{K}'$. The number of DAGs with 5 nodes for each configurations N_C is also shown. Notice that C_2 and C_3 have both zero counts and zero probability, because they are invalid.

(a)

$\mathbf{K}$	$\pi_{\Rightarrow}$	$\pi_{\Leftarrow}$	$\pi_{\Leftrightarrow}$	$\pi_{\not\Leftrightarrow}$
$r_{X,Y}(r_1)$	0.8	0.132	0.028	0.04
$r_{Y,Z}(r_2)$	0.9	0.066	0.014	0.02
$r_{X,Z}(r_3)$	0.6	0.264	0.056	0.08

$\mathbf{K}'$	$\pi_{\Rightarrow}$	$\pi_{\Leftarrow}$	$\pi_{\Leftrightarrow}$	$\pi_{\not\Leftrightarrow}$
$r_{X,Y}(r_1)$	0.764	0.159	0.032	0.045
$r_{Y,Z}(r_2)$	0.879	0.082	0.016	0.023
$r_{X,Z}(r_3)$	0.646	0.231	0.051	0.073

(b)

	$r_{X,Y}$	$r_{Y,Z}$	$r_{X,Z}$	J_C	N_C
C_1	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	0.6443	2800
C_2	$\Rightarrow$	$\Rightarrow$	$\Leftarrow$	0	0
C_3	$\Rightarrow$	$\Rightarrow$	$\Leftrightarrow$	0	0
...	...	...	...	...	...
C_{49}	$\not\Leftrightarrow$	$\Rightarrow$	$\Rightarrow$	$4.55 \cdot 10^{-4}$	1045
...	...	...	...	...	...
C_{64}	$\not\Leftrightarrow$	$\not\Leftrightarrow$	$\not\Leftrightarrow$	$2.78 \cdot 10^{-5}$	309

5 COMPUTING THE NUMBER OF DAGS N_C

The number N of DAGs over nodes $\mathcal{V}$ has been solved in closed-form [Robinson, 1973]. However, to the best of our knowledge, there is no closed-form for the number N_C of DAGs that satisfy certain path-constraints. When the number of nodes is small (up to 5-6) one can enumerate all DAGs and compute each N_C by counting. The number of possible DAGs however, grows exponentially to the number of nodes and complete enumeration is not an option. In this case, we estimate these counts by sampling a number S of DAGs uniformly at random. Specifically, we implemented the recent method in [Kuipers and Moffa, 2013] that, unlike prior work [Melancon et al., 2000], avoids the use of expensive MCMC methods. $\hat{N}_C$ can be estimated as $N \cdot S_C/S$, where S_C is the number of sampled DAGs that conform to configuration C.

When the number of configurations c is large or N_C/N is small, one may never sample any graph consistent with C, resulting in zero estimates. This may happen even for small sets of path variables, as c grows exponentially with the number of path variables n. To avoid zero estimates, one can apply the Laplace correction: $\hat{N}_C = \frac{S_C+l}{S+cl}N$, where l is an arbitrary parameter. We suggest l to be close to zero. Later on we will refer to this method as FULL$_l$.

In order to get a good estimate of N_C using FULL$_l$, one may have to sample a huge number of DAGs. To improve upon this we developed another method to approximate N_C. This method is based on the observation that, often, certain subsets of path variables are "almost independent". We exploit this to factorize the uninformative prior distribution U of each configuration, denoted with U_C for configuration C. N_C can then be computed as $U_C \cdot N$.

5.1 FACTORING THE UNINFORMATIVE PRIOR DISTRIBUTION U

To give an intuitive understanding of the main idea, consider the following scenario: we are given two path variables, $r_{X,Y}$ and $r_{W,Z}$. Notice that they do not have any nodes in common. Assume that we fix the value of $r_{W,Z}$. Depending on that value, some values of $r_{X,Y}$ will become more or less likely. For example, if $W \not\Leftrightarrow Z$ holds, the values $X \Rightarrow Y$, $X \Leftarrow Y$ and $X \Leftrightarrow Y$ become less likely since $W \not\Leftrightarrow Z$ restricts the graph to contain fewer edges, effectively reducing the possibility to form paths between X and Y. On the other hand, $X \not\Leftrightarrow Y$ becomes more likely. To put it formally, $U_{X \Rightarrow Y|W \not\Leftrightarrow Z} < U_{X \Rightarrow Y}$, $U_{X \Leftarrow Y|W \not\Leftrightarrow Z} < U_{X \Leftarrow Y}$, $U_{X \Leftrightarrow Y|W \not\Leftrightarrow Z} < U_{X \Leftrightarrow Y}$ and $U_{X \not\Leftrightarrow Y|W \not\Leftrightarrow Z} > U_{X \not\Leftrightarrow Y}$. However, we claim that *if the number of nodes $\mathcal{V}$ is sufficiently large*, the difference is negligible, or formally that $U_{r_{X,Y}|r_{W,Z}} \simeq U_{r_{X,Y}}$, that is, they are "almost independent". We illustrate this with a simple example. Assume that $\mathcal{V} = \{X, Y, W, Z\}$. In this case it is clear that any value of $r_{W,Z}$ heavily constrains the graph, since it only contains 4 nodes. If however we keep adding nodes to $\mathcal{V}$, more and more possibilities are created to satisfy any value of $r_{X,Y}$.

Next we show an example with dependent path variables. We are given the path variables $r_{X,Y}$ and $r_{Y,Z}$. Notice that Y appears in both path variables. Now consider the configurations $C_1 = \{X \Rightarrow Y, Y \Rightarrow Z\}$ and $C_2 = \{X \Rightarrow Y, Y \Leftarrow Z\}$. Note that the prior probability of a directed path between any two nodes is equal for any pair of nodes. Assuming U can be factorized, $U_{Y \Rightarrow Z|X \Rightarrow Y} = U_{Y \Rightarrow Z}$, and $U_{Y \Leftarrow Z|X \Rightarrow Y} = U_{Y \Leftarrow Z}$ hold. Because $U_{Y \Rightarrow Z} = U_{Y \Leftarrow Z}$ holds, $U_{Y \Rightarrow Z|X \Rightarrow Y} = U_{Y \Leftarrow Z|X \Rightarrow Y}$ follows. However, given $X \Rightarrow Y$, $Y \Rightarrow Z$ becomes less likely since there are no DAGs with $Z \Rightarrow X$ (acyclicity), which is not the case for $Y \Leftarrow Z$. For example, for $\mathcal{V} = \{X, Y, Z\}$ there are only 2 DAGs with configuration C_1, but 4 DAGs with configuration C_2. Thus U cannot be factorized in this case.

Those scenarios only give a rough and intuitive understanding of the basic idea. In the next subsection we will provide experimental results to support our claims. Before doing so, we will generalize the basic ideas to any set of path beliefs.

Definition 1. *Let* $\mathbf{R}$ *be a set of path variables. We denote with* V_R *the set of all nodes appearing in any variable in* $\mathbf{R}$*. The* ***constraint graph*** $G_R = (V_R, E_R)$ *of* $\mathbf{R}$ *is an undirected graph, where* $E_R = \{X - Y\}_{r_{X,Y} \in \mathbf{R}}$.

Definition 2. *Let* $\mathbf{R}$ *be a set of path variables and* $\mathbf{P}$ *a partition of* $\mathbf{R}$*. Let* $\mathcal{V}_{R_i}$ *denote the set of all nodes appearing in any variable in the i-th part of* $\mathbf{P}$, $\mathbf{P}_i$*.* $\mathbf{P}$ *is called an* ***independent partition*** *if* $\forall \mathbf{P}_i, \mathbf{P}_j \in \mathbf{P}, i \neq j, \mathcal{V}_i \bigcap \mathcal{V}_j = \emptyset$.

Since the parts of an independent partition do not have any nodes in common, the configuration of a part does not *directly* influence any other part; they do however have an *indirect* influence through other nodes of the graph which, as we will see, is negligible. On the other hand, path variables of the same part do directly affect each other (see dependent case above).

The independent partition of a set of path variables $\mathbf{R}$ can be computed as follows: (a) construct the constraint graph G_R of $\mathbf{R}$ and, (b) find the connected components of G_R. It is easy to see that the connected components of G_R are an independent partition of $\mathbf{R}$.

It remains to show how to compute U for given set of path variables $\mathbf{R}$ and set of nodes $\mathcal{V}$. First we sample S DAGs over $\mathcal{V}$ uniformly at random. Then we find an independent partition $\mathbf{P}$ of $\mathbf{R}$. U_C is factorized as $U_C = \prod_i U_{C_i}$, where C_i and U_{C_i} denote the configuration and the prior distribution of the i-th part $\mathbf{P}_i$ of $\mathbf{P}$ respectively. U_{C_i} is estimated as S_{C_i}/S, where S_{C_i} is the number of sampled DAGs that conform to configuration C_i in $\mathbf{P}_i$. Finally, $\hat{N}_C = N \cdot U_C$. Again, we recommend a Laplace correction. Then, $\hat{N}_C = N \cdot \frac{S \cdot U_C + l}{S + l \cdot c}$. We will refer to this method as FACT$_l$.

5.2 EXPERIMENTAL VALIDATION

The first experiment is to determine how FACT$_l$ approximates FULL$_l$, as the number of nodes $|\mathcal{V}|$ increases. We denote with U_{FACT_l} and U_{FULL_l} the estimation of U by the methods FACT$_l$ and FULL$_l$.

Setup: The number of nodes is varied between 10 and 35, with a step-size of 1. We used three sets of path variables: $R_1 = \{r_{1,2}, r_{3,4}\}$, $R_2 = \{r_{1,2}, r_{2,3}, r_{4,5}, r_{5,6}\}$, and $R_3 = \{r_{1,2}, r_{2,3}, r_{3,4}, r_{5,6}, r_{6,7}, r_{7,8}\}$. The number of independent partitions is 2 and each paritition consists of 1,2 and 3 path beliefs for R_1, R_2 and R_3 respectively. The number of valid configurations c is 16, 256 and 1681 for R_1, R_2 and R_3 respectively. The number of sampled DAGs S was set to 10^6, sufficiently large for FULL$_l$ to approximate U well. The Laplace correction parameter l was set to 0, since no correction is necessary in this case. We used the KL-divergence to measure the distance between two probability distributions, with U_{FULL_l} representing the true distribution.

Results: The results are shown in Figure 1a. As claimed, for a fixed set of path beliefs, U_{FACT_l} approaches U_{FULL_l} (which should be close to U in this experiment, as S is large relative to c) as the number of nodes increases. Similar results are expected with more and larger independent partitions.

In the second experiment we show that if S is relatively small compared to c, FACT$_l$ *provides a better approximation of* U *than* FULL$_l$. This is important because sampling a large number of DAGs costs time and memory, essentially setting an upper limit to S which, if c is relatively large, will result in a poor approximation of U by FULL$_l$. To show this, one has to know the exact distribution U. However, as this is computationally infeasible for large numbers of nodes, we ran the experiment only for small $|V|$.

Setup: The number of nodes is $|\mathcal{V}| = \{4, 5, 6\}$, and the number of DAGs is 543, 29281 and 3781503 respectively. Because $|\mathcal{V}|$ is small, we used only two path variables $\mathbf{R} = \{r_{1,2}, r_{3,4}\}$. For each $\mathcal{V}$ we sampled between 100 and 10000 DAGs, with a step-size of 100. This was done to simulate the case where no access to the complete set of DAGs is given. The Laplace correction constant l was set to 1. For each $|\mathcal{V}|$ and S we measured the KL-divergence between U_{FULL_l} and U, as well as between U_{FACT_l} and U. The experiment was repeated 1000 times and averages are reported.

Results: The results are shown in Figures 1b to 1d. When S is small, FACT$_l$ provides a better approximation of U than FULL$_l$. The reason this works is that, if $\mathbf{R}$ is partitioned into multiple sets, each containing a relatively small number of path variables, their distributions are easier to approximate.

6 COMPUTING THE J.P.D. J

In this section, we show how to compute the joint probability distribution J. We denote with $\pi_{k,j}$ the probability that r_k takes value $j \in \{\Rightarrow, \Leftarrow, \Leftrightarrow, \not\Leftrightarrow\}$: $\pi_{k,j} = P(r_k = j)$. The *unknown quantities* are J_C for each configuration C in J. Let $\mathcal{C}_{k,j} = \{C, \text{ s.t. } r_k = j\}$, that is, the set of configurations where variable r_k obtains value j. For each k and j we obtain the following constraints:

$$\pi_{k,j} = \sum_{C \in \mathcal{C}_{k,j}} J_C \tag{3}$$

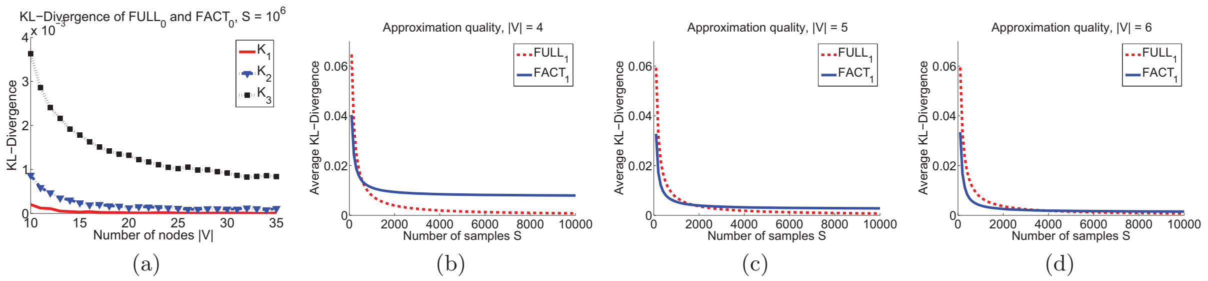


Figure 1: (a) KL-divergence between FULL_0 and FACT_0 with $S = 10^6$ for different sets of path variables. The distance between FACT_0 and the true distribution, approximated by FACT_0, decreases as the number of nodes increases. (b,c,d) KL-divergence between the true distribution and the approximation methods, as the number of samples S increases. For small S FACT_1 provides a better approximation of the true distribution than FULL_1.

In other words, the marginals of the j.p.d. should equal our input path beliefs. Recall that *path beliefs are not independent in general.* Thus, it is important to consider the following constraints, stemming from the path semantics of the variables $\mathbf{R}$:

$$J_C = 0, \text{ when } C \text{ is invalid} \tag{4}$$

A configuration is invalid if it cannot be satisfied by any DAG over $\mathcal{V}$, for example, it contains directed cycles. The algorithm to detect invalid configurations is discussed in Section 6.5. To complete the problem specification we impose that:

$$\sum_C J_C = 1 \quad \text{and} \quad J_C \geq 0 \tag{5}$$

If constraints in Eqs. 3, 4, 5 can be satisfied then a j.p.d. adhering to the probability axioms can be found such that the prior marginal beliefs hold. In this case, by definition, $\mathbf{K}$ is *coherent*, otherwise it is *incoherent*.

6.1 THE CASE OF COHERENT BELIEFS

The system of equations contains $4n$ constraints from Eq. 3, 1 constraint from Eq. 5 and $c = O(4^n)$ unknowns. For most typical problems, $4n + 1 \ll c$ and so the system may have infinite solutions. We argue that one should choose a solution j.p.d. J as close to the uninformative one as possible. Any other distribution may introduce bias towards certain configurations, even if the prior knowledge does not suggest preference over those configurations. In other words, if the uninformative j.p.d. U is a coherent extension of the path beliefs, there is no reason to prefer any other solution over it. A natural, information-theoretic approach is to select a j.p.d. J that minimizes the KL-divergence from U. The problem is formulated as:

$$\min_{\mathbf{J}} D_{KL}(J \parallel U) = \sum_{k=1}^{c} J_k \cdot \ln \frac{J_k}{U_k}, \text{ s.t. Eqs. 3, 5} \tag{6}$$

This optimization problem can be solved accurately and efficiently with the Iterative Scaling procedure [Darroch and Ratcliff, 1972, Csiszar, 1975], a generalization of the Iterative Proportional Fitting Procedure (IPFP) [Deming and Stephan, 1940].

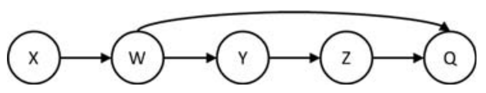
6.2 DEALING WITH INCOHERENT BELIEFS

In the case of incoherent beliefs there is no j.p.d. that can equal the marginal input beliefs. Instead of requesting coherent beliefs or ignoring the incoherency, we seek for joints with marginals as close as possible to the user's input beliefs. To solve this problem, we implemented the method proposed in [Vomlel, 2004], called GEMA. GEMA is an extension of IPFP which converges even with incoherent beliefs. In order to solve the problem it allows the marginals to change by a small amount, which is measured with the so-called I-aggregate criteria. Although GEMA tends to minimize this criteria, no guarantee about its convergence to a global or local minima is provided. We conducted some anecdotal experiments and GEMA seems to produce reasonable results.

Table 1b contains the j.p.d. J stemming from $\mathbf{K}$ of Table 1a(Top) computed by GEMA. For comparison with the input beliefs $\mathbf{K}$, Table 1a(Bottom) contains the marginal beliefs $\mathbf{K}'$ implied by GEMA. The values in Table 1a(Top) and Table 1a(Bottom) are close, with the latter one representing coherent beliefs. Figure 2 shows two DAGs with different configurations obtaining different prior scores.

6.3 FACTORING THE J.P.D. J

The cost of solving Eq. 6 is dominated by the number of variables c, which can be as high as 4^n. In practice, the optimization problem can not be solved efficiently (or at all, due to memory limitations) with more than


(a)

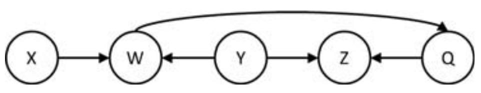

(b)

Figure 2: We assume the path beliefs $\mathbf{K}$ in Table 1a and the corresponding J in Table 1b. (a) The configuration $C_1 = \{X \Rightarrow Y, Y \Rightarrow Z, X \Rightarrow Z\}$ holds in the graph. For $p_1 = 0.6443$ we obtain the score $Sc(G_1|J) = \log(0.6443) - \log(2800) = -8.3769$. (b) The configuration $C_{49} = \{X \not\Leftrightarrow Y, Y \Rightarrow Z, X \Rightarrow Z\}$ holds in the graph. For $p_{49} = 4.55 \cdot 10^{-4}$ we obtain the score $Sc(G_2|J) = \log(4.55 \cdot 10^{-4}) - \log(1045) = -14.6471$. As expected, G_1 has a higher prior than G_2 since $X \Rightarrow Y$ is given a higher probability than $X \not\Leftrightarrow Y$ in Table 1a.

10-12 path beliefs. It is obvious that, even in the best case, one would need at least $\Omega(c)$ time and memory, if the output of the procedure is the full j.p.d. over c.

One natural way to improve upon this is to factorize J. Unfortunately, in general, it seems that it is not possible without loss of information. However, as stated in [Vomlel, 2004], if the uninformative joint distribution U factorizes with respect to some sets of variables, then the result of IPFP also factorizes with respect to the same sets of variables, that is, if $\exists \{R_i\}_{i=1}^{k}, R_i \subseteq \mathbf{R}$ s.t. $U = \prod_{R_i} U_{R_i}$ then $J = \prod_{R_i} J_{R_i}$. Thus, if we use FACT$_l$ instead of FULL$_l$ to compute U, we can usually compute J significantly faster and for larger sets of path beliefs, that is, instead of a total limit of 10-12 path beliefs, each part of the independent partition used in FACT$_l$ has a limit of 10-12 path beliefs.

6.4 ADJUSTING MISLEADING PRIORS

In practice, it may be the case that some priors are misleading, that is, the correct value of a path variable r has a lower probability than any other value of r. It is not always possible to detect those cases; however, it is possible to do so when the path beliefs are dependent, and the majority of them gives preference to the correct relation. We illustrate this with a simple example. Assume that the correct relation between two variables X and Y is $X \Rightarrow Y$, and that an expert suggests that $P(X \Rightarrow Y) = 0.1$. Now assume that we have path beliefs that $P(X \Rightarrow V) = P(V \Rightarrow Y) = 0.9$. They are incoherent: by the probability axioms, $P(X \Rightarrow Y) \geq 0.8$ follows. Our method will implicitly consider this and will increase $P(X \Rightarrow Y)$ while reducing $P(X \Rightarrow V)$ and $P(V \Rightarrow Y)$. The effect will be even higher if more path beliefs suggest that $P(X \Rightarrow Y)$ is high. For example, if $P(X \Rightarrow Y) = 0.1$ and we have 4 such pairs of path beliefs $P(X \Rightarrow V_i) = P(V_i \Rightarrow Y) = 0.9$, our method will assign $P(X \Rightarrow Y) = 0.632$ and $P(X \Rightarrow V_i) = P(V_i \Rightarrow Y) = 0.814\ \forall i$. We see that, although $P(X \Rightarrow Y)$ was low initially, it was given a high probability by our method because the other beliefs supported $X \Rightarrow Y$. Thus, *considering dependent beliefs and dealing with incoherence may identify and adjust misleading beliefs.*

6.5 INVALID CONFIGURATIONS

Let C be a configuration of path variables $\mathbf{R}$. C is **invalid** if $\exists r_{X,Y} \in \mathbf{R}$, s.t.: (a) $r_{X,Y} =$ " $\Rightarrow$ " and $r_{X,Y} =$ " $\Leftarrow$ " is implied by C (acyclicity), or (b) $r_{X,Y} =$ " $\Leftrightarrow$ " and $r_{X,Y} \in \{\Rightarrow, \Leftarrow\}$ is implied by C (definition of " $\Leftrightarrow$ "), or (c) $r_{X,Y} =$ " $\not\Leftrightarrow$ " and $r_{X,Y} \in \{\Rightarrow, \Leftarrow, \Leftrightarrow\}$ is implied by C (definition of " $\not\Leftrightarrow$ ").

These conditions are sufficient to identify invalid configurations, but not necessary. The simplest example is a dataset with two variables X and Y: the configuration $r_{X,Y} =$ " $\Leftrightarrow$ " is invalid as there is no other variable to serve as a common ancestor. Yet, the above cases will not identify it as such. However, when the number of variables in the data is large relative to the number of path variables (specifically if $|\mathcal{V}| \geq |V_R| + n$ holds)[1], these conditions are also necessary. *From now on we assume that the number of nodes in $\mathcal{V}$ is sufficiently large.*

7 SEARCH AND OPERATORS

In this paper we will use the **Greedy Search** method, searching in the space of DAGs. The method starts from a given initial DAG G_0 (usually chosen to be the empty DAG) and performs a hill-climbing search, considering all DAGs resulting by a **edge-insertion**, **edge-removal** or **edge-reversal** operation.

7.1 EXTENDING GREEDY SEARCH

Greedy Search can be trivially extended to additionally consider the prior score $Sc(G|J)$ of a DAG G. To do this, it first has to determine the configuration C_G of G, which can be computed in time $O(|V| \cdot n)$ given the transitive closure of G (stored as an adjacency matrix). The transitive closure of a DAG can be computed in time $O(|V|^2 + |V| \cdot |E|)$; run a DFS for each node and keep track of all visited nodes. There are faster and more complex algorithms [Simon, 1988], but the trivial method is usually faster for smaller graphs (we used the trivial method in our implementation).

[1]There are cases where a smaller number of variables is sufficient, but we did not further investigate it.

A problem is that, at each step of the search, the transitive closure has to be computed for all DAGs resulting by one of the search operators, whose number is $\Theta(|V|^2)$. The total cost is then $O(|V|^4 + |V|^3 \cdot |E| + |V|^3 \cdot n)$, which is a significant computational overhead. A straight-forward optimization is to dynamically update the closure after each edge insertion or removal. Various methods exist [Demetrescu and Italiano, 2008] trading off the time it takes to update the closure and querying for reachability. Assuming unit query time, a $O(|V|^2)$ update time is optimal [Demetrescu and Italiano, 2008]. Using this method, the time-complexity can be reduced to $O(|V|^4 + |V|^3 \cdot n)$.

7.2 SWAP-EQUIVALENT OPERATOR

To take advantage of the extra information provided by the path beliefs, one may have to use additional search-operators. That is because the standard operators make only small local improvements, without considering the global information provided by the path beliefs. Thus, an operator is desirable which is able to simultaneously make multiple adjustments in order to also change the configuration of the path variables.

We propose the **swap-equivalent-operator**. The idea is simple: at each step, after the application of a standard operator, we allow the algorithm to swap to a Markov equivalent DAG with the highest path belief score $Sc(G|J)$ increase. If the data score $Sc(G|D)$ has the score-equivalence property (e.g. BDe), the resulting DAG has the same data score but may have a higher prior score. This DAG can be computed with a simple modification of an algorithm presented in [Borboudakis and Tsamardinos, 2012]. Due to space limitations, the algorithm will not be described here.

8 EXPERIMENTAL RESULTS

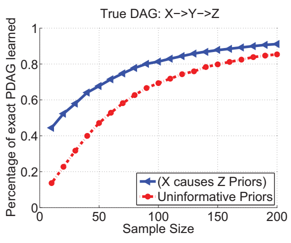
Employing Causal Knowledge. We consider the graph $X \rightarrow Y \rightarrow Z$. We use the path belief $P(X \Rightarrow Z) = 0.9$ and distribute the remaining 0.1 mass of probability to the remaining values of r_{XZ} proportional to the values that correspond to a uniform prior. We repeat the following experiment 10000 times: (a) we randomly select the number of states for each variable to be either 3 or 4, (b) we sample the cpts for each variable from the gamma distribution $\Gamma(k, \theta)$, with shape parameter k set to 0.5 and scale parameter θ set to 1, (c) we sample a dataset of size 200, (d) we increase the samples of the dataset provided to the scoring method from 10 to 200 with step size of 10, (e) we identify the highest scoring network out of all 25 possible DAGs using informative and uninformative priors and the BDeu score with Equivalent Sample Size (ESS) set to 1.

Results: Figure 3a plots the percentage of the time the PDAG $X - Y - Z$ of the true network was found exactly, with and without informative priors. First notice, that when the true PDAG is found, the edges are also *always oriented correctly since the true network has a higher prior than any other Markov-equivalent graph*. Perhaps more surprising though, notice that the informative priors also *improve the learning of the skeleton*. The belief $X \Rightarrow Z$ tends to add a path from X to Z. The associations $X - Y$ and $Y - Z$ are always higher than or equal to the association between $X - Z$ [Cover and Thomas, 2006]. Thus, *it is the correct path $X - Y - Z$ that tends to be induced, rather than any other network with a path $X \Rightarrow Z$.*

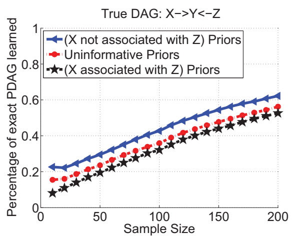
Employing Associative Knowledge. We run a similar proof-of-concept experiment where the true network is a single collider $X \rightarrow Y \leftarrow Z$. We use the same settings as before for three cases: correct associative priors $P(X \not\Leftrightarrow Z) = 0.9$, uniform priors, and incorrect associative priors $P(X \text{ associated with } Z) = 0.9$.

Results: The results are shown in Figure 3b. As expected, *correct prior beliefs clearly improve the chances of identifying the true PDAG; the effect is exactly the opposite when misleading, incorrect beliefs are provided to the algorithm.* Of course, asymptotically any non-zero priors play no role.

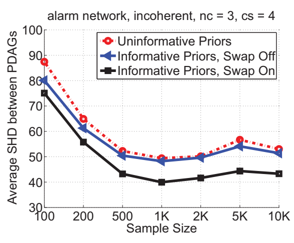
Learning Larger Networks. To generate path beliefs we use three parameters: the number of independent components nc, the number of nodes appearing in an independent component cs, and whether we want them to be coherent or incoherent. Path variables were generated as follows: for given cs and nc, we randomly pick nc non-overlapping sets, each containing cs nodes of the network, and consider all possible pairs between them as path variables, resulting in a total of $nc \cdot cs \cdot (cs-1)/2$ path variables. This is done in order to be able to consider large sets of path variables. Then, we randomly assign a probability $p \in [0.5, 0.99]$ to the true value of each path variable, and split the remaining $1 - p$ mass probability in an uninformative way to the remaining values. This process is repeated for each independent component until it is coherent or incoherent, depending on the input parameter. To estimate U we sampled $S = 10^6$ DAGs and l was set to the machine epsilon. We used the ALARM [Beinlich et al., 1989] and the INSURANCE [Binder et al., 1997] networks to evaluate our methods. We employed Greedy Search with the BDeu metric and ESS=1. We run the method starting from the empty graph with uninformative and informative priors, as well as with and without the swap-equivalent-operator in the case of informative priors. Finally, we compute the Structural Hamming Distance [Tsamardinos et al., 2006] from the PDAG of the true network. We used the PDAG to



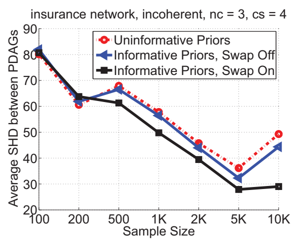
(a)



(b)



(c)



(d)

Figure 3: (a) Learning the orientations and the skeleton is facilitated by causal prior knowledge. (b) Learning the graph is facilitated by correct associative prior knowledge and hindered by incorrect priors. (c-d) Learning the ALARM and INSURANCE networks. The average Structural Hamming Distance (SHD) is shown with increasing sample size, for component size (cs) 4 and number of components (nc) set to 3, and incoherent beliefs. Using path beliefs, especially combined with the swap-equivalent operator, produces better networks on average.

avoid introducing an unfair advantage for our methods; all methods may find Markov equivalent DAGs, but the ones using path beliefs may find more correctly oriented edges. The sample size was varied within $\{100, 200, 500, 1000, 2000, 5000, 10000\}$. The path belief parameters were varied within $\{1, 2, 3, 4, 5\}$ and for nc and cs respectively, for both the coherent and incoherent cases. The experiment was repeated 100 times, for randomly sampled datasets and path beliefs, with all combinations of input parameters.

Results: Due to space limitations we report only the results for incoherent path beliefs, with $nc = 3$ and $sc = 4$ (18 beliefs). The results were similar for both, coherent and incoherent priors. Also, with smaller (larger) nc and sc, the difference between the uninformative and informative methods was smaller (larger).

The results are shown in Figures 3c and 3d. *In all cases, the SHD is smaller with the informative priors than with uninformative priors.* For the ALARM network, notice that the SHD difference between the uninformative method and the informative method without the operator decreases as sample size increases. The reason is that, as sample size increases, the data score becomes more important and the prior score tends to be ignored; it usually is considered only close to local maxima, where only small improvements in the data score can be made. If however the swap-equivalent operator is used, this does not happen, as it tries to maintain a high prior score during the whole search. Finally, notice the counter-intuitive behavior of increasing SHD with increasing sample size in Figure 3d for 10K samples. Anecdotal experiments suggest that the value of the ESS parameter is the reason for that behavior. However, when the swap-equivalent operator is used, this phenomenon is almost nonexistent.

9 CONCLUSIONS

We present a method for computing informative priors given a set of causal and associative beliefs on pairs of variables, as well as a novel search-operator to take advantage of them. The priors can then be employed by any search-and-score learning algorithm. The method, for the first time, addresses the issues of incoherent and possibly dependent priors. Providing correct priors about pairwise causal or associative relations improves learning both in terms of identifying the orientation of the edges (for causal priors), but also in terms of identifying the skeleton of the network.

There are numerous issues to still address regarding both the method and the general problem. The algorithm has exponential worst-case time complexity, thus more efficient algorithms are desirable. Closed-form solutions for computing the number of graphs given path constraints are also desirable. Finally, including other types of prior knowledge, as well as incorporating the strength of the causal effects or associations and other prior knowledge characteristics is an interesting future direction to pursue.

Acknowledgements

This work was partially funded by EU FP7 No 306000 STATegra, EU FP7 248590 REACTION, and Greek GSRT Thalis 20332 Symbiomics. We would like to thank the anonymous reviewers and Prof. G. F. Cooper for suggestions.

References

N. Angelopoulos and J. Cussens. Bayesian learning of Bayesian networks with informative priors. *Annals* *of Mathematics and Artificial Intelligence*, 54(1-3): 53–98, 2008.

I. Beinlich, G. Suermondt, R. Chavez, and G. Cooper. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. *Proceedings of the 2nd European Conference in Artificial Intelligence in Medicine*, pages 247–256, 1989.

J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2-3):213–244, 1997.

G. Borboudakis and I. Tsamardinos. Incorporating causal prior knowledge as path-constraints in Bayesian networks and maximal ancestral graphs. *Proceedings of the 29th International Conference on Machine Learning*, pages 1799–1806, 2012.

G. Borboudakis, S. Triantafillou, V. Lagani, and I. Tsamardinos. A constraint-based approach to incorporate prior knowledge in causal models. *Proceeding of the 19th European Symposium on Artificial Neural Networks*, pages 321–326, 2011.

W. Buntine. Theory refinement on Bayesian networks. *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence*, pages 52–60, 1991.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

T. M. Cover and J. A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2nd edition, 2006.

I. Csiszar. I-Divergence geometry of probability distributions and minimization problems. *Annals of Probability*, 3(1):146–158, 1975.

J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

C. Demetrescu and G. F. Italiano. Maintaining dynamic matrices for fully dynamic transitive closure. *Algorithmica*, 51(4):387–427, 2008.

W. E. Deming and F. F. Stephan. On a least squares adjustment of a sampled frequency table when the expected marginal totals are known. *Annals of Mathematical Statistics*, 11(4):427–444, 1940.

P. Hansen, B. Jaumard, M. Poggi de Aragão, F. Chauny, and S. Perron. Probabilistic satisfiability with imprecise probabilities. *International Journal of Approximate Reasoning*, 24(2-3):171–189, 2000.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

J. Kuipers and G. Moffa. Uniform generation of large random acyclic digraphs. *ArXiv e-prints*, May 2013.

C. Meek. Causal inference and causal explanation with background knowledge. *Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence*, pages 403–410, August 1995.

G. Melancon, M. Bousquet-Melou, and I. Dutour. Random generation of DAGs for graph drawing. Technical report, CWI, Stichting Mathematisch Centrum, 2000.

R. E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, Upper Saddle River, NJ, USA, 2003.

R. S. Niculescu, T. M. Mitchell, and R. B. Rao. Bayesian network learning with parameter constraints. *Journal of Machine Learning Research*, 7 (July):1357–1383, 2006.

R. T. O'Donnell, A. E. Nicholson, B. Han, K. B. Korb, M. J. Alam, and L. R. Hope. Incorporating expert elicited structural information in the CaMML causal discovery program. *Proceedings of the 19th Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence*, pages 1–16, 2006.

J. Pearl. *Causality, Models, Reasoning, and Inference*. Cambridge University Press, New York, NY, USA, 2000.

C. Robert and S. Arno. Priors on network structures. biasing the search for Bayesian networks. *International Journal of Approximate Reasoning*, 24(1):39–57, 2000.

R. W. Robinson. Counting labeled acyclic digraphs. *New Directions in the Theory of Graphs: Proceedings of the Third Annual Arbor Conference on Graph Theory*, pages 239–273, 1973.

K. Simon. An improved algorithm for transitive closure on acyclic digraphs. *Theoretical Computer Science*, 58(1-3):325–346, 1988.

P. Spirtes. Introduction to causal inference. *Journal of Machine Learning Research*, 11(May):1643–1662, 2010.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT Press, Cambridge, MA, 2nd edition, 2000.

I. Tsamardinos, L. Brown, and C. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

J. Vomlel. Integrating inconsistent data in a probabilistic model. *Journal of Applied NonClassical Logics*, 14(3):367–386, 2004.

SparsityBoost: A New Scoring Function for Learning Bayesian Network Structure

Eliot Brenner,[*] **David Sontag**
Courant Institute of Mathematical Sciences
New York University

Abstract

We give a new consistent scoring function for structure learning of Bayesian networks. In contrast to traditional approaches to score-based structure learning, such as BDeu or MDL, the complexity penalty that we propose is data-dependent and is given by the probability that a conditional independence test correctly shows that an edge cannot exist. What really distinguishes this new scoring function from earlier work is that it has the property of becoming computationally *easier* to maximize as the amount of data increases. We prove a polynomial sample complexity result, showing that maximizing this score is guaranteed to correctly learn a structure with no false edges and a distribution close to the generating distribution, whenever there exists a Bayesian network which is a perfect map for the data generating distribution. Although the new score can be used with any search algorithm, we give empirical results showing that it is particularly effective when used together with a linear programming relaxation approach to Bayesian network structure learning.

1 Introduction

We consider a fundamental problem in statistics and machine learning: how can one automatically extract structure from data? Mathematically this problem can be formalized as that of learning the structure of a Bayesian network with discrete variables. Bayesian networks refer to a compact factorization of a multivariate probability distribution, one-to-one with an acyclic graph structure, in which the conditional probability distribution of each random variable depends only on the values of its parent variables. One application of Bayesian network structure learning is for the discovery of protein regulatory networks from gene expression or flow cytometry data (Sachs *et al.* , 2005).

Existing approaches to structure learning follow two basic methodologies: they either search over structures that maximize the likelihood of the observed data (score-based methods), or they test for conditional independencies and use these to constrain the space of possible structures. The former approach leads to extremely difficult combinatorial optimization problems, as the space of all possible Bayesian networks is exponentially large, and no efficient algorithms are known for maximizing the scores. The latter approach gives fast algorithms but often leads to poor structure recovery because the outcomes of the independence tests can be inconsistent, due to sample size problems and violations of assumptions.

We formulate a new objective function for structure learning from complete data which obtains the best of both worlds: it is a score-based method, based predominantly on the likelihood, but it also makes use of conditional independence information. In particular, the new objective has a "sparsity boost" corresponding to the log-probability that a conditional independence test correctly shows that an edge cannot exist. We show empirically that this new objective substantially outperforms the previous state-of-the-art methods for structure learning. In particular, on synthetic distributions we find that it learns the true network structure with less than half the data and one tenth the computation.

The contributions of this paper are the introduction of this new scoring function, a proof of its consistency (we show polynomial sample complexity), and a carefully designed importance sampling algorithm for efficiently computing the confidence scores used in the objective. For both the proof of sample complexity and

[*]Current Affiliation: Search & Algorithms, Shutterstock Inc., New York

the importance sampling algorithm, we develop several new results in information theory, constructing precise mappings between a parametrization of distributions on two variables and mutual information, and characterizing the rate of convergence of various quantities relating to mutual information. We expect that many of the techniques that we developed will be broadly useful beyond Bayesian network structure learning.

2 Background

This paper considers the problem of learning Bayesian network structure from complete data (no hidden variables or unobserved factors). Let $\mathcal{X} = (X_1, X_2, \ldots, X_n)$ be a collection of random variables. For reasons that we explain in the next section, our results are restricted to the case when the variables X_i are binary, i.e. $\mathrm{Val}(X_i) = \{0, 1\}$. Formally, a Bayesian network over $\mathcal{X}$ is specified by a pair (G, P), where $G = (V, E)$ is a directed acyclic graph (DAG) satisfying the following conditions: the nodes V correspond to the variables $X_i \in \mathcal{X}$ and E is such that every variable is conditionally independent of its non-descendants given its parents. The joint distribution can then be shown to factorize as $P(x_1, \ldots, x_n) = \prod_{i \in V} P(X_i = x_i \mid X_{\mathrm{Pa}(i)} = x_{\mathrm{Pa}(i)})$, where $\mathrm{Pa}(i)$ denotes the parent set of variable X_i in the DAG G, and $x_{\mathrm{Pa}(i)}$ refers to an assignment to the parents.

A Bayesian network G is called an independence map (I-map) for a distribution P if all the (conditional) independence relationships implied by G are present in P. Going one step further, G is called a perfect map for P if it is an independence map and the conditional independence relationships implied by G are the *only ones* present in P. By ω_N (or in some contexts, Y_N) we denote a sequence of observations of the random variables $\mathcal{X}$, generated i.i.d. from an unknown Bayesian network (G, P), where G is a perfect map for P. The problem that we study is that of learning the Bayesian network structure and distribution (G, P) from the samples ω_N.

The simplest case of learning BN structure is when we have two random variables, which we will call X_A and X_B. There are only two nonequivalent BN structures:

$$\begin{aligned} &G_0 \colon X_A \qquad X_B \text{ (``disconnected'')},\\ &G_1 \colon X_A \longrightarrow X_B \text{ (``connected'')}. \end{aligned}$$

The structure learning problem in this case is to return, based on ω_N, a decision $X_A \perp\!\!\!\perp X_B$ (G_0) or $X_A \not\perp\!\!\!\perp X_B$ (G_1). In other words, in this case, the structure learning problem is strictly equivalent to one case of *hypothesis testing*, a well-studied and classic problem in statistics, specifically testing the hypothesis of whether X_A and X_B are independent.

In the case of three or more variables, the equivalence no longer holds in any strict sense. *Constraint-based approaches* use the results of conditional independence tests to infer the model structure. These methods solve the structure learning problem sequentially by first learning the undirected skeleton of the graph, $\mathrm{Skel}(G)$, and then orienting the edges to obtain a DAG. Assuming that G is a perfect map for P, if A is conditionally independent of B then we can conclude that neither $A \to B$ nor $B \to A$ can be in G. It can be shown that either A's parents or B's parents will be a separating set proving their conditional independence (there may be others). Thus, if we make the key assumption that each variable has at most a fixed number of parents d, then this can yield a polynomial time algorithm for structure learning (Spirtes *et al.* , 2001; Pearl & Verma, 1991). However, this approach has a number of drawbacks: difficulty setting thresholds, propagation of errors, and inconsistencies.

Let $p = p(\omega_N, A, B \mid s)$ denote the empirical distribution of A and B conditioned on an assignment $S = s$ for $S \subseteq V \backslash \{A, B\}$, and marginalized over all of the other variables. The *mutual information* statistic,

$$MI(p) = \sum_{a \in \mathrm{Val}(A),\, b \in \mathrm{Val}(B)} p(a, b|s) \log \left(\frac{p(a, b|s)}{p(a|s)p(b|s)} \right)$$

is a measure of the conditional independence of A and B conditioned on $S = s$. Given infinite data, two variables are independent if and only if their mutual information is zero. However, with *finite* data, mutual information is *biased* away from zero (Paninski, 2003). As a result, it can be very difficult to distinguish between independence and dependence.

An alternative approach is to construct a scoring function which assigns a value to every possible structure, and then to find the structure which maximizes the score (Lam & Bacchus, 1994; Heckerman *et al.* , 1995). Perhaps the most popular score is the BIC (Bayesian Information Criterion) score:

$$S_{\psi_1}(\omega_N, G) = LL(\omega_N|G) - \psi_1(N) \cdot |G|. \tag{1}$$

Here, $LL(\omega_N|G)$ is the log-likelihood of the data given G, $|G|$ is the number of parameters of G, and $\psi_1(N)$ is a weighting function with the property that $\psi_1(N) \to \infty$ and $\psi_1(N)/N \to 0$ as $N \to \infty$. When $\psi_1(N) := \frac{\log N}{2}$, the score, now called MDL, can be theoretically justified in terms of Bayesian probability. Intuitively, we can explain the BIC/MDL score as a log-likelihood regularized by a *complexity penalty* to keep fully connected models (with the most parameters) from always winning. Finding the structure which maximizes the score is known to be NP-hard (Chickering, 1996; Chickering *et al.* , 2004; Dasgupta, 1999). Heuristic algorithms have been proposed for maximizing this score, such as greedy hill-climbing (Chickering, 2002; Friedman *et al.* , 1999) and, more recently, by formulating the structure learning problem as an integer linear program and solving using branch-and-cut (Cussens, 2011; Jaakkola *et al.* , 2010).

The running time of solving the integer linear programs dramatically increases as the amount of data used for learning increases (see, e.g., Fig 4). This is counter-intuitive: more data should make the learning problem easier, not harder. The core problem is that as the amount of data increases, the likelihood term grows in magnitude whereas the complexity penalty shrinks. This is necessary to prove that these scoring functions are consistent, i.e. that in the limit of infinite data the structure which maximizes the score in fact is the true structure. As a consequence, however, the score becomes very flat near the optimum with a large number of local maxima, making the optimization problem extremely difficult to solve.

3 SparsityBoost: A New Score for Structure Learning

We design a new scoring function for structure learning that is both consistent and easy to solve regardless of the amount of data that is available for learning. The key property that we want our new scoring function to have is that as the amount of data increases, optimization becomes easier, not harder. When little data is available, it should reduce to the existing scoring functions.

Our approach is to add, to the BIC score, new terms derived from statistical independence tests. Before introducing the new score we provide some background on hypothesis testing. Let $\mathcal{P}$ denote the simplex of (joint) probability distributions over a pair of random variables, and let $\mathcal{P}_0$ denote the subset of product distributions: $\mathcal{P}_0 = \{q \in \mathcal{P} \mid MI(q) = 0\}$. For $q \notin \mathcal{P}_0$, the magnitude of $MI(q)$ provides a measure of how far q is from the set of product distributions. For $\eta > 0$, we define $\mathcal{P}_\eta := \{q \mid MI(q) \geq \eta\}$. The testing procedure has ω_N as input, null hypothesis H_0 (independence) for $p \in \mathcal{P}_0$, and alternative hypothesis H_1 for $p \in \mathcal{P}_\eta$. The *Type I error* α_N is defined as the probability of the test rejecting a true H_0, the *Type II error* β_N is defined as the probability of the test falsely accepting H_0, and the *power* is defined as $1 - \beta_N$.

The theory of Neyman-Pearson hypothesis testing for composite hypotheses tells us how to construct a hypothesis test of maximal *power* for any α_N (Hoeffding, 1965; Dembo & Zeitouni, 2009). In our setting, the test corresponds to computing $MI(\omega_N) := MI(p(\omega_N))$ and deciding on H_1 if the test statistic exceeds a threshold γ. Let $\beta_N(\gamma)$ denote the Type II error of the Neyman-Pearson test with threshold γ.

We propose using in our score the Type II error of the test with threshold $MI(\omega_N)$,

$$\beta_N^{p^\eta}(MI(\omega_N)) := \Pr_{Y_N \sim p^\eta}\{MI(Y_N) \leq MI(\omega_N)\},$$

where p^η is the M-projection of $p(\omega_N)$ onto $\mathcal{P}_\eta$, that is, with $H(\cdot\|\cdot)$ denoting the Kullback-Leibler divergence,

$$p^\eta := \operatorname*{argmin}_{p \in \mathcal{P}_\eta} H(p(\omega_N)\|p). \tag{2}$$

An intuitive explanation for the Type II error is that $\beta_N^{p^\eta}(\gamma)$ is the probability of obtaining a test statistic $MI(Y_N)$, $Y_N \sim p^\eta$, that is *more extreme*, in the *wrong* direction of independence, than the observed test statistic γ. On the one hand, if $\omega_N \sim p_0 \in \mathcal{P}_0$, then with high probability the power of the test with threshold $MI(\omega_N)$ approaches 1 and $\beta_N^{p^\eta}(MI(\omega_N))$ approaches 0, exponentially fast as $N \to \infty$; on the other hand, if $\omega_N \sim p_1 \in \mathcal{P}_\epsilon$, where $\epsilon > \eta$, then with high probability the power approaches 0 and $\beta_N^{p^\eta}(MI(\omega_N))$ approaches 1, as $N \to \infty$.

Now we can state our new score for structure learning and explain its remaining features:

$$S_{\eta,\psi_1,\psi_2}(\omega_N, G) = LL(\omega_N|G) - \psi_1(N)\cdot|G| + \psi_2(N)\cdot \sum_{(A,B)\notin G} \max_{S \in S_{A,B}(G)} \min_{s \in \text{val}(S)} -\ln\left[\beta_N^{p^\eta}(MI(p(\omega_N, A, B|s)))\right]$$

The first line is the BIC score. In the second line $\psi_2(N)$ is a weighting function such that $\psi_2(N)/N \to 0$ as $N \to \infty$: $\psi_2(N) := 1$ in the experiments. Each term in the sum is called a *sparsity boost*. The sum contains one sparsity boost for each *nonexistent* edge $(A, B) \notin G$. If $A \perp\!\!\!\perp B|(S = s)$, then the sparsity boost is $\Theta(N)$ as $N \to \infty$, and if $A \not\perp\!\!\!\perp B|(S = s)$, then it is $O(1)$, and further, in that case the sparsity boost becomes insignificant compared to the LL term (since $\psi_2(N)/N \to 0$).

Second, the sets $S_{A,B}(G)$, called *separating sets*, are certain subsets of the power set of $V - \{A, B\}$, which provide *certificates for statistical recovery of* G. More precisely, we have $(A, B) \notin G$, if and only if there is a *witness* $S \in S_{A,B}(G)$ such that $A \perp\!\!\!\perp B|S$. The most common ways of defining $S_{A,B}(G)$ are as follows:

$$S_{A,B}(G) = \{S \subset V\backslash\{A, B\} \mid |S| \leq d\}, \tag{3}$$
$$S_{A,B}(G) = \{\text{Pa}_G(A)\backslash B,\ \text{Pa}_G(B)\backslash A\}. \tag{4}$$

The family of assignments $(A, B, G) \mapsto S_{A,B}(G)$ for all (A, B) ranging over distinct pairs of vertices and G over some family $\mathcal{G}$ of DAGs, constitutes a *collection of separating sets*, denoted by $\boldsymbol{\mathcal{S}}$.

In order for $A \perp\!\!\!\perp B|S$ to hold, we must have $A \perp\!\!\!\perp B|s$, for *every* joint assignment $s \in \mathrm{Val}(S)$. This is the reason for taking the minimum over $s \in \mathrm{Val}(S)$ of the possible sparsity boosts. The existence of just *one* $S \in S_{A,B}(G)$ such that $A \perp\!\!\!\perp B|S$ suffices to rule out (A, B) as an edge in G. This is the reason for taking the maximum over $S \in S_{A,B}(G)$. The sparsity boost is $O(1)$ for an $(A, B) \in G$, and $\Theta(N)$ for an $(A, B) \notin G$.

It remains to explain how to compute $\beta_N^{p^\eta}(\gamma)$ in the implementation of the score S_{η,ψ_1,ψ_2}. According to the definition (2), p^η is data-dependent, and this makes it impractical to compute $\beta_N^{p^\eta}(MI(\omega_N))$ quickly enough for use in our algorithm. We make an approximation by fixing p^η to be a single "reference" distribution, with uniform marginals and satisfying $MI(p^\eta) = \eta$. In the case when $\mathrm{Val}(X_i) = \{0, 1\}$, there are two such distributions. Namely, let p^0 denote the uniform distribution, and let

$$p^0(t) = \begin{bmatrix} \frac{1}{4}+t & \frac{1}{4}-t \\ \frac{1}{4}-t & \frac{1}{4}+t \end{bmatrix} \text{ for all } t \in \left(-\frac{1}{4}, \frac{1}{4}\right). \quad (5)$$

Clearly, $p^0(t)$ has uniform marginals. Consider the function $MI(p^0(t))$ for $t \in \left(0, \frac{1}{4}\right)$. On this interval $MI(p^0(t))$ is positive, increasing, and has range $\left(0, MI\left(p^0\left(\frac{1}{4}\right)\right)\right)$. Thus for each η in the range, there is a unique parameter value t_η^+ such that $MI(p^0(t_\eta^+)) = \eta$. By symmetry, we also have $MI(p^0(-t_\eta^+))) = \eta$; fix

$$p^\eta := p^0(t_\eta^+). \quad (6)$$

We compute t_η^+ by a binary search in the interval $\left(0, \frac{1}{4}\right)$; by (5) and (6) this suffices to compute p^η, and has to be done only once during the algorithm's setup.

Having computed p^η, we can compute $\beta_N^{p^\eta}(\gamma)$ for many values of N, γ, and store them in a table. During the learning phase, we evaluate $\beta_N^{p^\eta}(MI(\omega_N))$ by interpolation. We explain more details in Sec. 5.

Related work. Our new score is similar to other "hybrid" algorithms that use both conditional independence tests and score-based search for structure learning, notably Fast 2010's Greedy Relaxation algorithm (Relax) and Tsamardinos *et al.* 2006's Max-Min Hill-Climbing (MMHC) algorithm. The MMHC algorithm has two stages, first using independence tests to construct a skeleton of the Bayesian network, and then performing a greedy search over orientations of the edges using the BDeu score. The Relax algorithm starts by performing conditional independence tests to learn constraints, followed by edge orientation to produce an initial model. After the first model has been identified, Relax uses a local greedy search over possible relaxations of the constraints, at each step choosing the single constraint which, if relaxed, leads to the largest improvement in the score. Both of these algorithms separate the constraint- and score-based approaches into two distinct steps, in contrast to our approach which directly incorporates the conditional independence tests as a term in the score itself.

The only other work that we are aware of that has studied the incorporation of reliability of independence tests in score-based structure search is de Campos (2006). Their objective function is very different from ours, comparing the empirical mutual information to its expected value assuming independence (using the χ^2 distribution). In contrast to de Campos's MIT score, the SparsityBoost score is consistent, provably able to recover the true structure.

Importance of using Type II error. To our knowledge, all previous approaches for Bayesian network structure learning use the Type I error α_N in assessing the reliability of an independence test, which is asymptotically given by the χ^2 distribution. A relatively high threshold needs to be specified in order to prevent the false rejection of independence and to correct for multiple hypothesis testing. One of our key contributions is to show how to use β_N, the Type II error. *Minimizing the Type II error is essential because we want to err on the side of caution*, only having a large sparsity boost if we are sure that the corresponding edge does not exist. Type I errors, on the other hand, can be corrected by the part of the objective corresponding to the BIC score. If we had instead used the Type I error probability within our score, it would have corresponded to a dependence boost rather than independence, and would be fooled if we failed to find a good separating set (e.g., for computational reasons).

4 Polynomial Sample Complexity of the SparsityBoost Score

4.1 Statement of Main Results

In this section, we prove the consistency of the SparsityBoost score. In order to state our main results, we need to define certain additional parameters. First, there is a (small) positive integer, d, which bounds the in-degree of all vertices in G. The family of BNs on n vertices satisfying this condition is called $\mathcal{G}^d$.

Second, we formalize the notion of the minimal edge strength ϵ in G. Define

$$S_{A,B}(\mathcal{G}^d) := \bigcup_{G \in \mathcal{G}^d} S_{A,B}(G).$$

Recall that the witness sets in $\mathcal{S}$ provide certificates for statistical recovery of G. We quantify the edge strength of $(A, B) \in G$ with respect to $S_{A,B}(\mathcal{G}^d)$, i.e.

the amount of dependence even after conditioning, by

$$\epsilon((A,B),S_{A,B}(\mathcal{G}^d)) := \min_{S\in S_{A,B}(\mathcal{G}^d)} \max_{s\in \mathrm{Val}(S)} MI(p(A,B|s))$$

Then, let $\epsilon = \epsilon(G) = \min_{(A,B)\in G} \epsilon((A,B),S_{A,B}(\mathcal{G}^d))$.

Next, we need the notion of an *error tolerance* $\zeta > 0$, which in turn follows from a notion of a $G' \in \mathcal{G}^d$ being ζ-far from the true network (G,P). For any $G' \in \mathcal{G}^d$, define the probability distribution $p_{G',P}$ over $\mathcal{X}$ to be the distribution which factors according to G' and minimizes the KL-divergence from P, i.e.

$$p_{G',P} := \operatorname*{argmin}_{Q\,:\,G' \text{ is an I-map for } Q} H(P\|Q).$$

We call $H(P\|p_{G',P})$ the **divergence of $\boldsymbol{P}$ from $\boldsymbol{G'}$,** and if $H(P\|p_{G',P}) > \zeta$ we say that **$\boldsymbol{G'}$ is $\boldsymbol{\zeta}$-far from $\boldsymbol{(G,P)}$**. In Theorem 1(a) we set an **error tolerance of $\boldsymbol{\zeta}$**, which is to say that we specify that our learning algorithm should rule out all G' which are ζ-far from (G,P).

Finally, we need m, the (maximum) inverse probability of an assignment to a separating set. More precisely, for any $A,B \in V^2$, $A \neq B$, and $S \in S_{A,B}(\mathcal{G}^d)$, let $m_P(S) := \max_{s\in\mathrm{Val}(S)}[P(S=s)]^{-1}$. Then let

$$m = m_P(G,\mathcal{G}^d,\mathcal{S}) = \max_{(A,B)\in G} \max_{S\in S_{A,B}(\mathcal{G}^d)} m_P(S). \quad (7)$$

For all $(A,B) \notin G$, there will be at least one witness $S \in S_{A,B}(G)$ such that $A \perp\!\!\!\perp B|S$. Let

$$\hat{S}_P((A,B),G) := \operatorname*{argmin}_{S\in S_{A,B}(G)\,:\,A\perp\!\!\!\perp B|S} m_P(S).$$

Finally, let

$$\hat{m}_P(G,\mathcal{S}) := \max_{(A,B)\notin G} m_P(\hat{S}_P((A,B),G)). \quad (8)$$

Theorem 1 *Suppose that $(G,P) \in \mathcal{G}^d$ is a Bayesian network of n **binary** random variables and G is a perfect map for P. Set $S_{A,B}(G) = \{S \subset V\backslash\{A,B\} \mid |S| \leq d\}$. Assume that $(G,P) \in \mathcal{G}^d$ has minimal edge strength $\epsilon > 0$, and minimal assignment probabilities m, as defined in (7) and $\hat{m}_P(G,\mathcal{S})$, as defined in (8). Fix $\eta = \lambda\epsilon$ for $\lambda \in (0,1)$, an error probability $\delta > 0$, and a tolerance $\zeta > 0$. Let S_η denote our score S_{η,ψ_1,ψ_2} for $\psi_1(N) := \kappa\log(N)$ and $\psi_2(N) = 1$. Let ω_N be a sequence of observations sampled i.i.d. from P.*

(a) There is a function $N(\epsilon,m,n;\delta,\zeta;\eta,\kappa)$ in

$$\tilde{O}\left(\max\left(\frac{\log(n)m}{\epsilon^2},\ \frac{n^2}{\zeta^2}\right)\log\frac{1}{\delta}\right)$$

as $\epsilon,\zeta,\delta \to 0^+$, $n,m \to \infty$, such that for all $N > N(\epsilon,m,n;\delta,\zeta;\eta,\kappa)$, with probability $1-\delta$, we have

$$S_\eta(G,\omega_N) > S_\eta(G',\omega_N),$$

for all $G' \in \mathcal{G}^d$ which are ζ-far from G.

(b) Then there is a function $N(\epsilon,m,\hat{m}_P,n;\delta;\eta,\kappa)$ in

$$\tilde{O}\left(\max\left(\frac{\log(n)m}{\epsilon^2},\ \frac{n^2\hat{m}_P^2}{\epsilon^2}\right)\log\frac{1}{\delta}\right)$$

as $\epsilon,\delta \to 0^+$, $n,m,\hat{m}_P \to \infty$, such that for all $N > N(\epsilon,m,\hat{m}_P,n;\delta;\eta,\kappa)$, with probability $1-\delta$, we have

$$S_\eta(G,\omega_N) > S_\eta(G',\omega_N),$$

for all $G' \in \mathcal{G}^d$ such that $\mathrm{Skel}(G') \not\subseteq \mathrm{Skel}(G)$.

In order to explain the significance of this result, it is helpful to relate it to three representative sample complexity results in the literature: Höffgen (1993), Friedman & Yakhini (1996), Zuk *et al.* (2006). The result of Zuk *et al.* differs from the other two and from our result because it only gives conditions for the (BIC) score of G to beat that of an individual competing network G', not a family, such as $\mathcal{G}^d$. The main difference between Höffgen and Friedman & Yakhini is that, like our result, Höffgen assumes that the competing network lies in $\mathcal{G}^d$ and achieves a sample complexity that is polynomial in $n = \mathrm{card}(V)$, while Friedman & Yakhini puts no restriction on the in-degree of competing networks, and obtains complexity that is exponential in n. Our result and Zuk *et al.* differ from both Höffgen and Friedman & Yakhini in that we provide guarantees for learning the correct DAG structure G (or at least a G without false edges), not just a distribution P' which is ζ-close to P. For this reason, only our paper and Zuk *et al.* need to define a minimal edge strength as a parameter, whereas for Höffgen and Friedman & Yakhini the main parameter is the error tolerance ζ, which they call ϵ.

4.2 Overview of Proofs

The proof of Theorem 1 consists of showing that for all sufficiently large N we can find a (probable) lower bound on the *score difference*,

$$S_\eta(G,\omega_N) - S_\eta(G',\omega_N), \quad G,G' \in \mathcal{G}^d,\ \omega_N \sim G. \quad (9)$$

The score difference breaks down into a sum of the following terms:

(a) The difference of log-likelihood terms, $LL(G,\omega_N) - LL(G',\omega_N)$.

(b) The difference of complexity penalties, $\kappa\log(N)(|G'| - |G|)$.

(c) For each distinct pair of vertices $A,B \in V$ such that *neither* G nor G' has (A,B) as an edge, the difference of the sparsity boosts in the objective functions of G and G', for that nonexistent edge.

(d) For each *true edge* $(A,B) \in G$ missing from G', the negative of the sparsity boost for $(A,B) \notin G'$.

(e) For each *false edge* $(A,B) \notin G$ present in G', the (positive) sparsity boost for $(A,B) \notin G$.

With the choice of $\mathcal{S}$ in the Theorem, $S_{A,B}(G') = S_{A,B}(G)$ for all $A, B \in V^2$, which implies that (c) is exactly 0. Furthermore, (b) is clearly $O(\log N)$ for $G, G' \in \mathcal{G}^d$, while both (a) and (e) will turn out to be $\Theta(N^\alpha)$ for $\alpha > 0$, so that (b) has only minor impact on the sample complexity.

So we will focus on how to estimate (a), (d), and (e). Conceptually, estimating each of these terms calls for the same *type of result*: a *concentration lemma* stating how quickly the empirical $LL(\cdot, \omega_N)$ (for (a)), respectively $MI(\omega_N, A, B|s)$ (for (d) and (e)) converges to the "ideal" counterpart $LL^{(I)}(\cdot, P)$, respectively $MI(P, A, B|s)$. In fact, both of the latter consist of a *polynomial* in n number of terms (which is where we use the hypothesis $G \in \mathcal{G}^d$) of the form $p \log p$ for parameters p of certain Bernoulli random variables.

Proposition 1 *Let $p \in (0,1)$ be given and $X(p)$ the Bernoulli random variable with parameter p. Let $\hat{\epsilon}, \delta \geq 0$ be given. For $Y_N \sim p$, denote the empirical parameter p_{Y_N} by $\tilde{p}_N$. Then there is a function*

$$N(\hat{\epsilon}, \delta) \in O\left(\left(\frac{1}{\hat{\epsilon}}\right)^2 \log \frac{1}{\delta}\right), \tag{10}$$

as $\hat{\epsilon}, \delta \to 0^+$ with the property that for $N > N(\hat{\epsilon}, \delta)$,

$$\Pr\left(|\tilde{p}_N \log \tilde{p}_N - p \log p| < \hat{\epsilon}\right) \geq 1 - \delta.$$

Proposition 1 improves slightly on Lemma 1 in Höffgen (1993), by replacing $\tilde{O}(\cdot)$ with $O(\cdot)$ in (10).

A key feature of Proposition 1, for obtaining our concentration results for LL and MI is that (10) is independent of the Bernoulli parameter p. From the concentration result for LL, we can show that (a) is with high probability positive and larger than $N\zeta/3$, for all G' which are ζ-far from G and for sufficiently large N. From the concentration result for MI we can show that a sparsity boost from a *true* edge is bounded above by a constant for sufficiently large N (linear in m). So the negative contribution of (d) is bounded. These bounds suffice to prove Theorem 1(a).

In the proof of Theorem 1(b), from the concentration result for LL, we can show that (a) is with high probability larger than a constant times $-n\sqrt{\log(n)N}$. Furthermore, a sparsity boost from a *false* edge is $\Omega(\Gamma(\eta)N)$, where the speed $\Gamma(\eta)$ of the linear growth is on the order of η^2 as $\eta \to 0^+$. To show the latter, we first apply Proposition 1, given a witness, to prove that $MI(\omega_N, A, B|s)$ is (likely) less than $\eta/2$. Second, using a Chernoff bound, we show that $-\log \beta_N^{p^\eta}(\gamma)$ is $\Omega(\eta^2 N)$ for γ less than $\eta/2$. So, with high probability the positive contribution of (e) eventually overwhelms any negative contribution of (a).

The techniques derived from the Chernoff bound yield a version of Theorem 1(b) with an exponent of 4 on the ϵ in the denominator of the term $n^2\hat{m}_P^2/\epsilon^2$. To improve the exponent to 2, we need a strengthened result on the linear growth of a sparsity boost from a false edge, in which the speed $\Gamma(\eta)$ is on the order of only η instead of η^2, as $\eta \to 0^+$.

We have to use a new method derived from Sanov's Theorem instead of Chernoff's Bound. To our knowledge, the way we use Sanov's Theorem to study the concentration of mutual information is a novel contribution to information theory. For all of the following we are assuming that $\mathrm{Val}(X_i) = \{0,1\}$ for all $X_i \in V$ so that $\mathcal{P}$ is the space of probability distributions over the alphabet $\{0,1\}^2$. We have already, in (5), given a parameterization of the path of distributions of uniform marginals in $\mathcal{P}$. We now generalize (5) and the associated parameterization by defining

$$p(p_{A,0}, p_{B,0}, t) := \begin{bmatrix} p_{A,0}p_{B,0} + t & p_{A,1}p_{B,0} - t \\ p_{A,0}p_{B,1} - t & p_{A,1}p_{B,1} + t \end{bmatrix} \tag{11}$$

where $p_{A,1} := 1 - p_{A,0}$ and $p_{B,1} := 1 - p_{B,0}$. When $(p_{A,0}, p_{B,0})$ ranges over $[0,1]^2$ and t over $(t_{\min}, t_{\max})$ (an interval depending on $p_{A,0}$, $p_{B,0}$), (11) parameterizes the whole space $\mathcal{P}$.

Since the t parameter is a measure of how far p is from $\mathcal{P}_0$, it is not surprising that we can derive formulas relating t to $\sqrt{MI}$. In order to carry this out, we consider the function $MI(p(p_{A,0}, p_{B,0}, t))$ as a function of t and carefully study the Taylor series expansion of this function around the basepoint $t = 0$.

The reason for preferring the t parameter to MI itself is that by means of Sanov's Theorem and Pinsker's Inequality, we obtain a very general result which bounds $-\log \beta_N^{p^\eta}(\gamma)$ from below by N times the squared L_∞-distance of p^η from a distribution q^γ. More specifically, defining the complement of P_γ by

$$A_\gamma := \{p \in \mathcal{P} \mid MI(p) \leq \gamma\}, \tag{12}$$

the distribution q^γ is defined as the I-projection of p^η onto A_γ. We would like to relate $\|p^\eta - q^\gamma\|_\infty$ to $|MI(p^\eta) - MI(q^\gamma)| = |\eta - \gamma|$, and the t-parameters act as an effective intermediary, because it is easy to show that $\|p^\eta - q^\gamma\|_\infty$ is on the order of $|t_\eta^+ - t_\gamma^+|$, where t_γ^+ is the t-parameter of q^γ. Applying the relation of the preceding paragraph between t and $\sqrt{MI}$, we obtain a bound, from below, of $-\log \beta_N^{p^\eta}(\gamma)$ by something on the order of $(\sqrt{\eta} - \sqrt{\gamma})^2 N$.

5 Computation of β values

Exact computation. Here we give an exact formula for $\beta_N^{p^\eta}(\gamma)$ using the Method of Types (Cover & Thomas, 2006, Chapter 11). Denoting the entries of $p^\eta \in \mathcal{P}$ by $(p_{0,0}, p_{0,1}, p_{1,0}, p_{1,1})$, we have

$$\beta_N^{p^\eta}(\gamma) = \sum_{Y_N} \prod_{i,j=0}^{1} p_{i,j}^{T_{i,j}(Y_N)} \, \mathbf{1}[MI(Y_N) \le \gamma],$$

where $T_{i,j}(Y_N)$ is the number of observations of (i,j) in the sampled sequence Y_N of length N. Consider the set $\mathcal{T}_N$ of length-4 vectors of nonnegative integers $(T_{0,0}, T_{0,1}, T_{1,0}, T_{1,1})$ summing to N. Every $T \in \mathcal{T}_N$ corresponds one-to-one with a distribution $p_T \in \mathcal{P}$ (obtained by dividing every entry in T by N). Let $|T|$ denote the number of sequences Y_N corresponding to type T. Then it is not difficult to see that $|T|$ is given by a multinomial coefficient and that

$$\beta_N^{p^\eta}(\gamma) = \sum_{T \in \mathcal{T}_N} |T| \prod_{i,j=0}^{1} p_{i,j}^{T_{i,j}} \mathbf{1}_{A_\gamma}(p_T), \qquad (13)$$

where $\mathbf{1}_{A_\gamma}$ is the characteristic function of A_γ (see Eq. 12). We can use (13) to exactly compute $\beta_N^{p^\eta}(\gamma)$, but because of the summation over $\mathcal{T}_N$ the running time of this algorithm is $O(N^3)$, which will not scale to the range of N we need for our experiments.

Monte Carlo computation. In place of exact calculation, we estimate $\beta_N^{p^\eta}$ by means of Monte Carlo integration, using importance sampling of the domain to reduce the variance. In order to implement this, we first observe that (13) is essentially a Riemann sum for a definite integral, so that we may replace the summation with an integral. Second, the integrand we initially obtain in this manner has numerous discontinuities, because of the $|T|$ factor. It makes the next steps easier to implement if we replace $|T|$ with a (slightly larger) continuous approximation (Csiszar & Körner, 2011, p. 39). We finally obtain the following integral which approximates $\beta_N^{p^\eta}(\gamma)$ given in (13):

$$\left(\frac{N}{2\pi}\right)^{(|X|-1)/2} \int_{\mathcal{P}} e^{-NH(q\|p^\eta)} \prod_{i,j=0}^{1} q_{i,j}^{-\frac{1}{2}} \; \mathbf{1}_{A_\gamma}(q)\, \mathrm{d}q.$$

For the Monte Carlo integration we use an importance sampling scheme based on the following idea: the integrand is largest when $H(q\|p^\eta)$ is small and $q \in A_\gamma$, and so it should be strongly concentrated around q^γ (the I-projection of p^η onto A_γ). We have an unproven conjecture, supported by numerical evidence, that $q^\gamma = p^\gamma := p^0(t_\gamma^+)$ (the unproven part of this is that q^γ has uniform marginals) for η less than approximately 0.1109. The importance sampling algorithm samples points $p \in \mathcal{P}$ i.i.d., favoring points near p^γ. We use the parameterization (11) of $\mathcal{P}$ and sample the parameters $p_{A,0}, p_{B,0}$ & t independently according to Gaussian distributions. For the selection of the two marginals, we use identical Gaussians centered at $\frac{1}{2}$ and becoming more concentrated (exponentially fast) around their mean as $N \to \infty$. For the t parameter, we use use a third Gaussian centered at t_γ^+. For each (N, γ) we determine the concentration of the third Gaussian by sampling the integrand along the path $p\left(\frac{1}{2}, \frac{1}{2}, t\right)$, in the segment $(0, t_\gamma^+)$.

Since we cannot possibly tabulate $\beta_N^{p^\eta}(\omega_N)$ for every empirical sequence that might arise, we tabulate $\beta_N^{p^\eta}(\gamma)$ for N, γ in a strategically chosen grid of values, and during the learning phase we interpolate or extrapolate (as the need arises) from these tabulated values. We interpolate/extrapolate $-\ln \beta_N^{p^\eta}(\gamma)$ *linearly* in the statistics N and $H(p^\gamma\|p^\eta)$. Sanov's Theorem gives heuristic support to this procedure, but ultimately our justification for this procedure rests on the empirical results presented in Section 6 below.

6 Experimental Results

Computing the confidence measure. In Figure 1 we present several empirical results that help to justify our methods for calculating $\beta_N^{p^\eta}(\gamma)$, our new measure of the reliability of an independence test. First, in (a), we show that using the method of summing over types to calculate $\beta_N^{p^\eta}(\gamma)$ has a running time which is $O(N^4)$, whereas the Monte Carlo method explained in Section 5 is $O(1)$ as $N \to \infty$. Thus, although it is feasible to pre-compute $\beta_N^{p^\eta}(\gamma)$ for small values of N, exact calculation is impractical for N much larger than 200.

As for the accuracy of the Monte Carlo estimation, the table in Figure 2 shows that for very small N, e.g. $N < 50$, some multiplicative errors for our method of $\approx 30\%$ are observed, but by the time we reach $N = 100$, the errors are consistently $< 10\%$. Figure 1(b) shows that, for $N = 200$, the Monte Carlo estimate has a consistently small error over the range of γ.

The linear interpolation procedure for obtaining $\ln \beta_N^{p^\eta}(MI(\omega_N))$ from the pre-computed tables of $\ln \beta_N^{p^\eta}(\gamma)$ receives heuristic support from Sanov's Theorem; it receives empirical support from Figure 1(c) (resp. (d)), which shows that the dependence of $\ln \beta_N^{p^\eta}(\gamma)$ on N (resp., $H(p^\gamma\|p^\eta)$), assuming all other inputs are fixed, is roughly linear.

Sample Complexity. In this section we study the accuracy of our learning algorithm as a function of the amount of data we provide it. We compare our algorithm to two baselines: BIC and Max-Min Hill-Climbing. BIC is equivalent to our score without

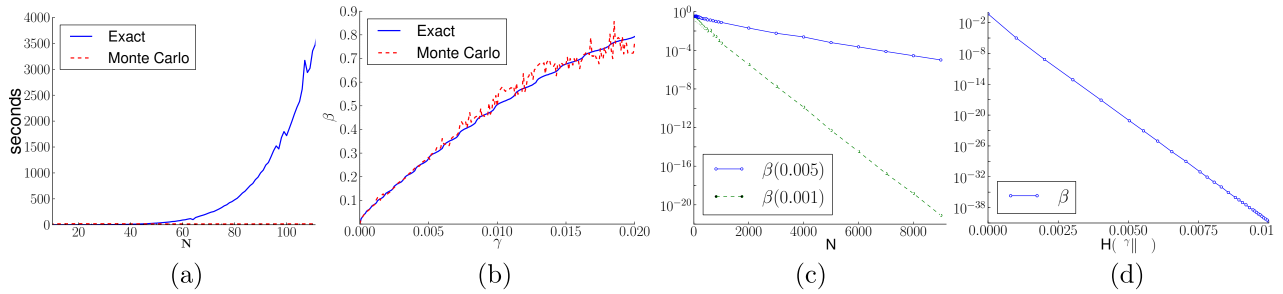


Figure 1: Computation of $\beta_N^{p^\eta}$. All results shown are for $\eta = 0.01$. (a) Running time of the exact algorithm to compute $\beta_N^{p^\eta}$ grows cubically in N, but for Monte Carlo approximation remains constant (results shown for $\gamma = 0.005$ and 0.001 combined). (b) Monte Carlo estimate of $\beta_N^{p^\eta}(\gamma)$ for fixed η, $N = 200$. (c) Exponential decay of $\beta_N^{p^\eta}(\gamma)$ in N for fixed γ. (d) Exponential decay of $\beta_N^{p^\eta}(\gamma)$ as a function of KL-divergence $H(p^\gamma \| p^\eta)$, as γ is varied, for large $N = 9000$.

	γ			γ	
N	0.001	0.005	N	0.001	0.005
20	12.20%	29.15%	70	1.03%	3.59%
30	24.65%	1.18%	80	9.06%	4.13%
40	39.77%	7.07%	90	0.77%	4.05%
50	2.45%	4.88%	100	1.01%	0.03%
60	3.52%	0.30%	110	2.27%	3.01%

Figure 2: Multiplicative error of Monte Carlo approximation, $|\beta_N^{p^\eta} - \hat{\beta}_N^{p^\eta}|/\beta_N^{p^\eta}$, for $\eta = 0.01$ as N, γ vary.

the sparsity boost terms. MMHC is state-of-the-art in terms of both speed and quality of recovery, and has been shown to outperform most other constraint-based approaches (Tsamardinos *et al.* , 2006). As we discussed earlier, MMHC is also a hybrid algorithm, using both conditional independence tests and score-based search. We use the implementation of MMHC provided by the authors as part of Causal Explorer 1.4 (Aliferis *et al.* , 2003), with the default parameters.[1]

We use an integer linear program to exactly solve for the Bayesian network that maximizes the BIC or SparsityBoost scores (Jaakkola *et al.* , 2010; Cussens, 2011). To solve the ILP, we use Cussens' GOBNILP 1.2 software together with SCIP 3.0 (Achterberg, 2009). Conveniently, since the sparsity boost terms in our objective can be subsumed into the parent set scores, we can use these off-the-shelf Bayesian network solvers without any modification.

The data that we use for learning is sampled from synthetic distributions based on the Alarm network structure (Beinlich *et al.* , 1989). The Alarm network has 37 variables, 46 edges, and a maximum in-degree of 4. In our synthetic distributions, every variable has only two states, and its conditional probability distribution is given by a logistic function, $p(X_i = 1 \mid \mathbf{x}_{\mathrm{Pa}(i)}) = 1/(1 + e^{-\vec{\theta}_i \cdot \mathbf{x}_{\mathrm{Pa}(i)} - u_i})$. We sampled 10 different distributions, with parameters drawn according to $\theta_{ij} \sim U[-.5, .5] + \frac{1}{4}\mathcal{N}(0,1)$ for $j \in \mathrm{Pa}(i)$ and $u_i \sim \frac{1}{4}\mathcal{N}(0,1)$. For each value of N, a new set of N samples were drawn from the corresponding synthetic distribution. The results shown are the average for each of these 10 synthetic distributions.

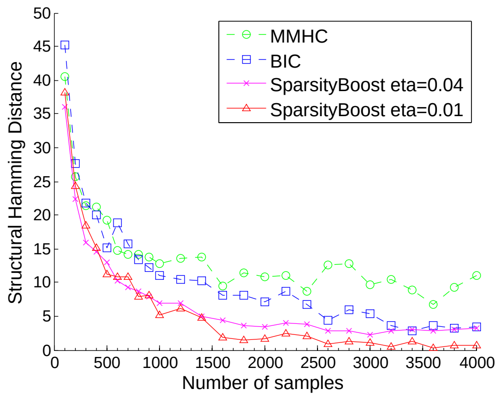


Figure 3: Comparison of the sample complexity of MMHC, BIC, and our new SparsityBoost objective. Each point is the average of the SHD of the learned network from truth for 10 synthetic distributions.

We use $S_{A,B}(G)$ from Eq. 3 with $d = 2$, enumerating over all separating sets of size at most two. Larger separating sets are less useful because they lead to a smaller ϵ, less data, and more computation to create the objective. In the Alarm network, for every $(A, B) \notin G$ there is a separating set S such that $|S| \leq 2$ and $A \perp B | S$. Regardless, if a separating set for an inexistent edge cannot be found, our score simply reduces to the BIC score, so no harm is done.

Our results are shown in Figure 3. We measure the quality of structure recovery using the Structural Hamming Distance (SHD) between the partially directed acyclic graphs (PDAG) representing the equivalence classes of the true and learned networks (Tsamardinos *et al.* , 2006). The SHD is defined as the number

[1]Threshold for χ^2 test of .05 and Dirichlet hyperparameters equal to 10. Varying these did not improve results.

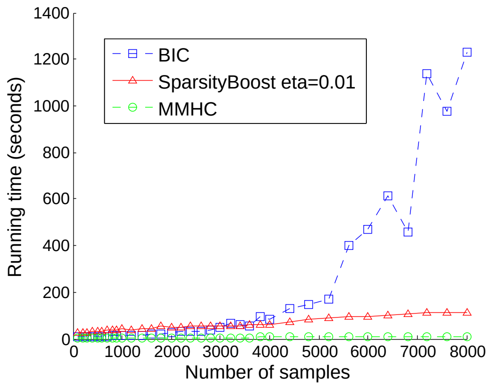


Figure 4: Total running time to learn a Bayesian network from data for BIC, SparsityBoost, and MMHC. We maximize the BIC and SparsityBoost scores by solving an integer linear program to optimality.

of edge additions, deletions, or reversals to make the two PDAGs match. The plots for SparsityBoost with $\eta = 0.005$ and $\eta = 0.02$ (not shown) are nearly identical to that of $\eta = 0.01$. SparsityBoost consistently learns better structures than MMHC or BIC, and often perfectly recovers the networks after only 1600 samples. SparsityBoost obtains a smaller average error with 3000 samples than BIC does with 6000, representing a more than 50% reduction in the number of samples needed for learning. We also found that the SparsityBoost results had substantially less variance than either BIC or MMHC.

Our theoretical results only guarantee exact recovery when $\eta < \epsilon$. For each of the synthetic distributions we computed $\epsilon(A, B)$ for all of the edges (A, B) in the true structure (see Sec. 4.1 for definition). The minimum of these, that is to say ϵ, ranged from .000028 to .0047, which is in fact *smaller* than the largest value of η considered in our experiments (.005). Despite this, we obtained excellent empirical results for SparsityBoost with $\eta \in \{.005, .01, .02\}$. This may be partially explained by the *average* value of $\epsilon(A, B)$ being .062. Even when we push η to be as high as .04, SparsityBoost converges to an average SHD of at most 3 (see Fig. 3). Thus, our new objective appears to be particularly robust to choosing the wrong value of η.

Running Time. We show the running time of our new objective compared to BIC in Figure 4. The figure shows the total time, which includes both the time to compute the score of all parent sets and the time to solve the ILP to optimality. These results confirm our hypothesis that the new score would be substantially easier to optimize. We found that the linear programming relaxation for SparsityBoost (with $\eta = 0.01$) was tight on *nearly all* instances: branch-and-bound did not need to be performed. Once the SparsityBoost objective has been computed, the ILP is solved within 6 seconds in every single instance.

The timing experiments reported in this section were performed on a single core of a 2.66 Ghz Intel Core i7 processor with 4 GB of memory. MMHC's average running time was less than 8 seconds for all sample sizes. MMHC is significantly faster because it quickly prunes edges that cannot exist and in its second step uses a greedy (rather than exact) optimization algorithm for score-based search.

7 Discussion

Our approach maintains the advantages of other score-based approaches to structure learning, such as the ability to find the K-best Bayesian networks and ease of introducing additional constraints (e.g., from interventional data). In order to optimize our score, virtually any optimization procedure can be used. Since the ILP is easy to solve, this suggests that greedy structure search may also be able to easily find the best-scoring Bayesian network under the SparsityBoost score.

One subject for future investigations is to generalize and sharpen our results in various ways. Using a similar construction for p^η, we believe it should be possible to extend our score and proof of consistency to non-binary variables. We also believe it will be possible to eliminate the dependence of $N(\epsilon, m, \hat{m}_P, n; \delta, \zeta; \eta, \kappa)$ in both parts of Theorem 1 on the parameter m, leaving only the dependence on $\hat{m}_P$ in part (b), which is in some cases much smaller than m.

Another issue to be explored as a future line of investigation is the choice of p^η in our measure of reliability, $\beta_N^{p^\eta}(\gamma)$. The overall motivation for $\beta_N^{p^\eta}(\gamma)$ is to capture the probability of Type II error of a statistical test with independent distributions $\mathcal{P}_0$ as the null hypothesis H_0 and all distributions $\mathcal{P}_\eta$ as the alternative hypothesis H_1. The choice of uniform marginals for p^η represents an expedient choice, providing an objective function that is manageable to implement and compute, yet still has a reasonable theoretical and empirical sample complexity. Better results might be obtained by setting the marginals of p^η to approximate those of $p(\omega_N)$. More generally, one can contemplate incorporating various other statistically derived probabilities into the objective function.

This leads to the broader point that objective functions, and the optimization of them over discrete spaces of structures, are ubiquitous throughout computer science and statistics. Our work suggests a new paradigm for incorporating information from "classical" hypothesis tests into the objective functions used for machine learning. This new paradigm provides both computational *and* statistical efficiency.

References

Achterberg, Tobias. 2009. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, **1**(1), 1–41.

Aliferis, Constantin F., Tsamardinos, Ioannis, Statnikov, Alexander R., & Brown, Laura E. 2003. Causal Explorer: A Causal Probabilistic Network Learning Toolkit for Biomedical Discovery. *Pages 371–376 of: METMBS.*

Beinlich, I. A., Suermondt, H. J., Chavez, R. M., & Cooper, G. F. 1989. The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks. *Pages 247–256 of: Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine*. Springer-Verlag.

Chickering, D. 1996. Learning Bayesian Networks is NP-Complete. *Pages 121–130 of:* Fisher, D., & Lenz, H.J. (eds), *Learning from Data: Artificial Intelligence and Statistics V*. Springer-Verlag.

Chickering, D. 2002. Learning Equivalence Classes of Bayesian-Network Structures. *Journal of Machine Learning Research*, **2**, 445–498.

Chickering, D., Heckerman, D., & Meek, C. 2004. Large-Sample Learning of Bayesian Networks is NP-Hard. *Journal of Machine Learning Research*, **5**, 1287–1330.

Cover, T.M., & Thomas, J.A. 2006. *Elements of information theory.* John Wiley & Sons.

Csiszar, I., & Körner, J. 2011. *Information theory: coding theorems for discrete memoryless systems.* Cambridge University Press.

Cussens, James. 2011. Bayesian network learning with cutting planes. *Pages 153–160 of: Proceedings of the Twenty-Seventh Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-11)*. Corvallis, Oregon: AUAI Press.

Dasgupta, S. 1999. Learning polytrees. *In: Proc. of the 15th Conference on Uncertainty in Artificial Intelligence.*

de Campos, Luis M. 2006. A Scoring Function for Learning Bayesian Networks based on Mutual Information and Conditional Independence Tests. *J. Mach. Learn. Res.*, **7**(Dec.), 2149–2187.

Dembo, A., & Zeitouni, O. 2009. *Large deviations techniques and applications.* Vol. 38. Springer.

Fast, A. 2010. *Learning the structure of Bayesian networks with constraint satisfaction.* Ph.D. thesis, University of Massachusetts Amherst.

Friedman, N., & Yakhini, Z. 1996. On the Sample Complexity of Learning Bayesian Networks. *In: Proc. of the 12th Conference on Uncertainty in Artificial Intelligence.*

Friedman, Nir, Nachman, Iftach, & Pe'er, Dana. 1999. Learning Bayesian Network Structure from Massive Datasets: The "Sparse Candidate" Algorithm. *Pages 206–215 of: UAI.*

Heckerman, D., Geiger, D., & Chickering, D. M. 1995. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, **20**(3), 197–243. Available as Technical Report MSR-TR-94-09.

Hoeffding, W. 1965. Asymptotically optimal tests for multinomial distributions. *The Annals of Mathematical Statistics*, 369–401.

Höffgen, K.U. 1993. Learning and robust learning of product distributions. *Pages 77–83 of: Proceedings of the sixth annual conference on Computational learning theory.* ACM.

Jaakkola, Tommi, Sontag, David, Globerson, Amir, & Meila, Marina. 2010. Learning Bayesian Network Structure using LP Relaxations. *Pages 358–365 of: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AI-STATS)*, vol. 9. JMLR: W&CP.

Lam, Wai, & Bacchus, Fahiem. 1994. Learning Bayesian Belief Networks: An Approach Based on the MDL Principle. *Computational Intelligence*, **10**, 269–294.

Paninski, Liam. 2003. Estimation of entropy and mutual information. *Neural Comput.*, **15**(6), 1191–1253.

Pearl, Judea, & Verma, Thomas. 1991. A Theory of Inferred Causation. *Pages 441–452 of: KR.*

Sachs, Karen, Perez, Omar, Pe'er, Dana, Lauffenburger, Douglas A., & Nolan, Garry P. 2005. Causal Protein-Signaling Networks Derived from Multiparameter Single-Cell Data. *Science*, **308**(5721), 523–529.

Spirtes, P., Glymour, C., & Scheines, R. 2001. *Causation, Prediction, and Search, 2nd Edition.* The MIT Press.

Tsamardinos, I., Brown, L. E., & Aliferis, C. F. 2006. The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm. *Machine Learning*, **65**(1), 31–78.

Zuk, Or, Margel, Shiri, & Domany, Eytan. 2006. On the Number of Samples Needed to Learn the Correct Structure of a Bayesian Network. *In: UAI.* AUAI Press.

Sample Complexity of Multi-task Reinforcement Learning

Emma Brunskill
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

Lihong Li
Microsoft Research
One Microsoft Way
Redmond, WA 98052

Abstract

Transferring knowledge across a sequence of reinforcement-learning tasks is challenging, and has a number of important applications. Though there is encouraging empirical evidence that transfer can improve performance in subsequent reinforcement-learning tasks, there has been very little theoretical analysis. In this paper, we introduce a new multi-task algorithm for a sequence of reinforcement-learning tasks when each task is sampled independently from (an unknown) distribution over a finite set of Markov decision processes whose parameters are initially unknown. For this setting, we prove under certain assumptions that the per-task sample complexity of exploration is reduced significantly due to transfer compared to standard single-task algorithms. Our multi-task algorithm also has the desired characteristic that it is guaranteed not to exhibit negative transfer: in the worst case its per-task sample complexity is comparable to the corresponding single-task algorithm.

1 INTRODUCTION

A dream of artificial intelligence is to have lifelong learning agents that learn from prior experience to improve their performance on future tasks. Our interest in the present paper is in how to transfer knowledge and improve performance across a sequence of reinforcement-learning [Sutton and Barto, 1998] problems, where each task itself involves sequential decision making under uncertainty in an unknown environment. We assume that each task is drawn from a finite set of Markov decision processes with identical state and action spaces, but different reward and/or transition model parameters; however, the MDP parameters are initially unknown and the MDP identity of each new task is also unknown. This model is sufficiently rich to capture important applications like tutoring systems that teach a series of students whose initially unknown learning dynamics can be captured by a small set of types (such as honors, standard and remedial), marketing systems that may characterize a customer into a finite set of types and use that to adaptively provide targeted advertising over time, and medical decision support systems that seek to provide good care to patients suffering from the same condition for whom the best treatment strategy may be characterized by a discrete hidden latent variable that captures the patient's physiology.

Although there is encouraging empirical evidence that transferring information across tasks can improve reinforcement learning performance (see Taylor and Stone [2009] for a recent survey), there has been almost no theoretical work to justify or quantify the benefits. This is highlighted as one of the key limitations of the existing research by Taylor and Stone [2009], and there have been only a few papers since then that provide any theoretical analysis [Lazaric and Restelli, 2011, Mann and Choe, 2012]. In particular, we are aware of no work that seeks to formally analyze how transferred knowledge can accelerate reinforcement learning in a *multi-task* settings.

In contrast, there has been a substantial amount of interest over the last decade on Probably Approximately Correct (PAC) reinforcement learning in the single-task setting (e.g. [Kearns and Singh, 2002, Brafman and Tennenholtz, 2002]). This line of work formally quantifies the worst-case learning speed of a reinforcement-learning algorithm, defined as the number of steps in which the agent may fail to follow an ϵ-optimal policy.

In this paper, we introduce a new algorithm for multi-task reinforcement learning, and prove under certain assumptions that the per-task sample complexity is significantly reduced due to transfer compared to the single-task sample complexity. Furthermore, unlike most prior multi-task or transfer reinforcement learning algorithms, our proposed algorithm is guaranteed to avoid negative transfer: the decrease in performance that can arise when misleading information is transferred from a source to target task.

2 PRELIMINARIES

This paper focuses on discrete-time, finite Markov decision processes (MDPs) defined as a tuple $\langle S, A, P, R, \gamma \rangle$, where S is the finite state space, A the finite action space, P is the transition probability function, R the reward function, and $\gamma \in (0, 1)$ the discount factor. The reward function is bounded and without loss of generality takes values in $[0, 1]$. For convenience, we also use S and A to denote the cardinality of the state and action spaces, respectively.

A deterministic policy $\pi : S \to A$ defines what action to take in a given state. Its value function, $V^\pi(s)$, is defined as the expected total discounted reward received by executing π starting from state $s \in S$. Similarly, the state–action value function, $Q^\pi(s, a)$, is defined as the expected total discounted reward received by taking action a in state s and following π thereafter. It is known [Puterman, 1994] that there exist optimal value functions satisfying: $V^* = \max_\pi V^\pi$ and $Q^* = \max_\pi Q^\pi$; furthermore, the greedy policy with respective to Q^* is optimal.

Typically, a reinforcement-learning (RL) [Sutton and Barto, 1998] agent does not know the transition probability and reward functions, and aims to optimize its policy via interaction with the MDP. The main objective of RL is to approximate an optimal policy with as few interactions as possible. One formal framework for analyzing the speed of learning in RL, which we adopt here, is the *sample complexity of exploration* [Kakade, 2003], or *sample complexity* for short. Fix parameters $\epsilon > 0$ and $\delta > 0$. An RL algorithm $\mathbf{A}$ can be viewed as a nonstationary policy whose value functions can be defined similarly to stationary policies π above. At any timestep h, we compare the policy value to the optimal policy value in the current state s_h. If $V^*(s_h) - V^{\mathbf{A}_h} > \epsilon$, then $\mathbf{A}$ is not near-optimal in timestep h, and a mistake happens. If, with probability at least $1 - \delta$, the total number of mistakes made by an algorithm is at most $\zeta(\epsilon, \delta)$, then ζ is called the sample complexity of exploration. RL algorithms with a polynomial sample complexity is called PAC-MDP. Further details and related works are found in the survey of Strehl et al. [2009].

In this paper, we consider multi-task RL across a series of T reinforcement-learning tasks, each run for H steps. We assume each task is sampled from a set $\mathcal{M}$ of C MDPs, which share the same state and action spaces, and discount factor, but have different reward and/or transition dynamics. Finally, we denote by $V_{\max}$ an upper bound of the value function. Note that $V_{\max} \le 1/(1 - \gamma)$, but can be much smaller than this upper bound in many problems.

3 PAC-MDP MULTI-TASK RL

We are interested in exploring whether it is possible to reduce sample complexity when the agent faces a sequences of tasks drawn i.i.d. from a distribution, and if so, how this benefit can be achieved by an algorithm.

Algorithm 1 Multi-task RL Algorithm

```
0: Input: $T_1, \bar{C}$.
1: for $t = 1, 2, \ldots, T_1$ do
2:     Receives an unknown MDP $M_t \in \mathcal{M}$
3:     Run $E^3$ in $M_t$ for $H$ steps to get counts $o(s, a, s', t)$
4: end for
5: Combine counts into $\hat{C} \le \bar{C}$ groups where
   $\bar{o}(s, a, s', c)$ is the counts for the $c$-th group.
6: for $t = T_1 + 1, \ldots, T$ do
7:     Receive unknown $M_t \in \mathcal{M}$.
8:     Run Finite-Model-RL on $M_t$
9:     if MDP group of task $M_t$ is identified then
10:        Incorporate state–action visitation counts from
           $M_t$ to the group.
11:    end if
12: end for
```

Prior work suggests that higher performance is achievable when there is some known structure about the RL MDP parameters. In particular, past research has shown that when a task is drawn from a known distribution over a known finite set MDPs, the problem can be cast as a partially observable MDP planning problem, and solved to yield the Bayes optimal solution if the set cardinality is small [Poupart et al., 2006, Brunskill, 2012]. Although the past work did not examine the sample complexity of this setting, it does suggest the possibility of significant improvements when this structure can be leveraged.

Encouraged by this work, we introduce a two-phase multi-task RL algorithm. Since at the beginning the agent does not know the model parameters, it does single-task learning, and uses the observed transitions and rewards to estimate the parameters of the set of underlying MDPs at the end of phase one. In the second phase, the agent uses these learned models to "accelerate" learning in each new task. We will shortly provide details about both phases of this algorithm, whose performance is formally analyzed in the next section. Before doing so, we also note that our multi-task RL algorithm is designed to minimize or eliminate the potential of negative transfer: tasks where the algorithm performs much worse than a single-task RL algorithm.

Compared to Bayesian approaches, our algorithm development is motivated and guided by sample-complexity analysis. In addition to the guard against negative transfer, our approach is robust, as guaranteed by the theory; this benefit can be shown empirically even in a toy example.

In the following discussion, for clarity we first present a slightly simplified version of our approach (Algorithm 1), before discussing a few additional details in Section 3.3 that involve subtle technicalities required in the analysis.

Algorithm 2 Finite-Model-RL

```
0: Input: $S$, $A$, $\bar{o}$, $\hat{C}$, $m$, $\xi$, $\epsilon$.
1: Initialize the version space: $\mathcal{C} \leftarrow \{1, \ldots, \hat{C}\}$.
2: $\forall 1 \le i < j \le \hat{C}$: $c_{ij} \leftarrow 0$, $\Delta_{ij} \leftarrow 0$,
3: $\forall s, a : o(s, a) \leftarrow 0$ (Initialize counts in current task)
4: KNOWN $\leftarrow$ Check-Known($S, A, \epsilon, \bar{o}, \mathcal{C}, o, m$)
5: Use $E^3$ algorithm to compute an explore-or-exploit
   policy and the corresponding value function.
6: Initialize start state $s$
7: for $h = 1, \ldots, H$ do
8:    Take action $a$, receive reward $r$, and transition to the
      next state $s'$
9:    for all $c \in \mathcal{C}$ do
10:      Predict the model dynamics by empirical means:
         $\hat{\theta}_c \leftarrow \langle \hat{p}(s_1|s, a, c), \ldots, \hat{p}(s_{|S|}|s, a, c), \hat{r}_c \rangle$
11:      Compute the $\ell_2$-confidence interval of $\hat{\theta}_c$ (by, say,
         Lemma 5); denote the confidence interval by $\delta\theta_c$.
12:   end for
13:   Encode the transition $(s, a, r, s')$ by a vector (where
      $\mathbf{I}$ is the indicator function)
      $z \leftarrow \langle \mathbf{I}(1 = s'), \mathbf{I}(2 = s'), \ldots \mathbf{I}(|S| = s'), r \rangle$
14:   for all $i, j \in \mathcal{C}$ such that $i < j$ and $\left\| \hat{\theta}_i - \hat{\theta}_j \right\| \ge$
      $8 \max_{c \in \mathcal{C}} \delta\theta_c$ do
15:      $c_{ij} \leftarrow c_{ij} + \frac{1}{4} \left\| \hat{\theta}_i - \hat{\theta}_j \right\|^2$
16:      $\Delta_{ij} \leftarrow \Delta_{ij} + \left\| \hat{\theta}_i - z \right\|^2 - \left\| \hat{\theta}_j - z \right\|^2$
17:      if $c_{ij} \ge \xi$ then
18:         $\mathcal{C} \leftarrow \mathcal{C} \setminus \{c\}$ where $c = i$ if $\Delta_{ij} > 0$ and $c = j$
            otherwise.
19:      end if
20:   end for
21:   KNOWN $\leftarrow$ Check-Known($S, A, \epsilon, \bar{o}, \mathcal{C}, o, m$)
22:   If KNOWN has changed, use $E^3$ to re-compute the
      policy.
23: end for
```

3.1 PHASE ONE

In the first phase, T_1 tasks are drawn i.i.d. from the underlying unknown distribution. On each task, the agent follows the single-task algorithm E^3 [Kearns and Singh, 2002]. All observed transitions and rewards are stored for each task.

At the end of the first phase, this data is clustered to identify a set of at most $\bar{C}$ MDPs. To do this, the transition and reward parameters are estimated by the empirical means for each task, and tasks whose parameters differ by no more than a fixed threshold are clustered together. After this clustering completes, all the observed transitions and rewards for all tasks in the same cluster are *merged* to yield a single set of data. Our analysis below shows that, under certain assumptions, tasks corresponding to the same MDP will be grouped correctly.

Algorithm 3 Check-Known

```
0: Input: $S$, $A$, $\epsilon$, $\bar{o}$, $\mathcal{C}$, $o$, $m$.
1: for $(s, a) \in S \times A$ do
2:    if All MDPs in $\mathcal{C}$ have $\epsilon$-close (in $\ell_2$-norm) estimates
      of the transition and reward functions at $(s, a)$ then
3:       KNOWN$(s, a) \leftarrow$ true
4:    else if $o(s, a) \ge m$ then
5:       KNOWN$(s, a) \leftarrow$ true
6:    else
7:       KNOWN$(s, a) \leftarrow$ false
8:    end if
9: end for
```

3.2 PHASE TWO

At the start of phase two, the agent now has access to a set of (at most) $\bar{C}$ MDPs which approximate the true set of MDP models from which each new task is sampled. The key insight is that the agent can use these candidate models to identify the model of the current task, and then act according to the policy of identified model, and that this process of model identification is generally faster than standard exploration needed in single-task learning.

To accomplish this, we introduce a new single-task RL algorithm, Finite-Model-RL (Algorithm 2), that draws upon but extends the noisy-union algorithm of Li et al. [2011]. One critical distinction is that our approach can be used to compare models that themselves do not have perfect estimates of their own parameters, and do so in way that allows us to eliminate models that are sufficiently unlikely to have generated the observed data.

Like many single-task PAC-MDP RL algorithms, Finite-Model-RL partitions all state–action pairs into known and unknown, where a known state–action pair is one for which we have an ϵ-accurate estimate of its parameters. Following E^3, Finite-Model-RL maintains two MDP models for the present task. In the *exploration MDP*, the algorithm assigns unity rewards to unknown states and zero rewards to others. This MDP will be useful for computing a policy to explore unknown states. The other MDP, called the *exploitation MDP*, is identical to the underlying MDP except for unknown states, where rewards are all zero and state transitions are self-loops. This MDP is used to exploit existing knowledge about the current MDP in order to follow a reward-maximization policy.

Similar to E^3, our algorithm prioritizes exploration over exploitation: if the optimal value function of the current state in the exploration MDP is above a threshold, the estimated exploration MDP's optimal policy is followed; otherwise, the estimated exploitation MDP's optimal policy is used. In addition, Finite-Model-RL tracks which of the possible set of $\hat{C}$ MDP models could be the underlying MDP of the current task. It eliminates a model when there is sufficient evidence in the observed transitions that it is not the true model (see lines 14–21 in Algorithm 2). We do this by tracking the difference in the sum of the ℓ_2 error between the current task's observed (s, a, s', r) transitions and the transitions predicted given each of the $\hat{C}$ MDP models obtained at the end of phase 1.

A particular state–action pair becomes known when either there are sufficient observations from the current task that the parameters of that state–action pair can be accurately estimated (as in single-task PAC-MDP RL), or if the remaining possible MDP models have ϵ-close estimates of the state–action pair's parameter in question. If the MDP of the current task is identified (only a single model remains possible in the set), then all the observed data counts from that MDP can be merged with the current task observed data counts. This frequently causes all state–action pairs to become immediately known, and then the algorithm switches to exploitation for the remainder of the task. At the end of each task, the underlying MDP will be identified with high probability, and the observed counts from the current task will be added to the counts for that MDP.

Since the base algorithm in Finite-Model-RL is very much like E^3, we preserve the standard single-task sample-complexity guarantee. Thus, negative transfer is avoided in any single task in phase 2, [1] compared to single-task E^3.

Across tasks, the observations accumulate and the $\hat{C}$ MDP models will eventually have ϵ-accurate estimates of all state–action parameters. Once this occurs, when facing a new task, as soon as the agent identifies the task model out of the $\hat{C}$ candidates, all state–action pairs become known. We will shortly see that the sample complexity for this identification to occur can be much smaller than standard single-task sample complexity bounds.

3.3 ADDITIONAL ALGORITHM DETAILS

We now describe a few additional algorithmic details that have been avoided on purpose to make the main ideas clear. In our analysis later, as well as in some practical situations, these technical details are important.

The key additional detail is that the E^3 algorithm is run with two different knownness threshold parameters at different stages of the multi-task algorithm: this parameter specifies the accuracy on the parameter estimates required for a state–action pair to be considered known in standard E^3. Usually, this parameter is set to $O(V_{\max}^2/(\epsilon^2(1-\gamma)^2))$ so that once a state–action is known, its dynamics can be estimated sufficiently accurately. However, in multi-task RL, we also need to identify task identity in order to facilitate knowledge transfer to benefit future task.

This observation motivates the use of two different values for this parameter. At the beginning of a task, one may want to use a relatively small value just to do a more balanced random walk in the whole state space, with the primary goal to identify the present task by visiting "informative" states. Here, a state is informative if two MDP models have a sufficient disagreement in its reward or transition dynamics; formal details are given in the next section. Only after the identity is known does the algorithm switch to a larger value, on the order of $O(V_{\max}^2/(\epsilon^2(1-\gamma)^2))$, to learn a near-optimal policy. If, on the other hand, the learner chooses the large value as specified in single-task PAC-MDP algorithms, it is possible that the learner does not visit informative states often enough by the end of a task to know its identity, and the samples collected cannot be transferred to benefit solving future tasks.

More precisely, in phase 1, we first execute E^3 with knownness threshold $O(\Gamma^{-2})$, where Γ, to be defined in the next section, measures the model discrepancy between two MDPs in $\mathcal{M}$, and is in general much larger than ϵ. Once E^3 has finished its exploration phase (meaning all state–action pairs have $O(\Gamma)$-accurate parameter estimates), we switch to running E^3 with the regular threshold of $O(V_{\max}^2/(\epsilon^2(1-\gamma)^2))$. Since E^3 performs all exploration before commencing exploitation, and $\epsilon < \Gamma$, the sample complexity of the resulting method stays the same as initially running E^3 with an input ϵ parameter. This ensures that we maintain the single-task sample-complexity guarantees, but also that we gain enough samples of each state–action pair so as to reliably cluster the tasks at the end of phase 1. With the same approach in phase 2, we can ensure that the task will be identified (with high probability).[2]

Finally, we note that information can also be transferred to the current task through tighter optimistic bounds on the value function that shrink as models are eliminated. Briefly, in phase 2, we can compute an upper bound $\bar{Q}_i$ of the state–action values of the $i \in \hat{C}$ MDPs that also accounts for any uncertainty in the model parameters. At each step, the value of each unknown state–action pair (s, a) can then be set to $\max_{i \in \mathcal{C}} \bar{Q}_i(s, a)$. Since this modification does not seem to impact the worst-case sample complexity, for clarity we did not include it in the description of Algorithm 2, although it may lead to practical improvement.

4 ANALYSIS

This section provides an analysis of our multi-task RL algorithm. As mentioned in Section 3.3, two values are used to define the knownness threshold in E^3. Due to space limitation, some of the proof details are left to a full version.

To simplify exposition, we use θ_i to denote MDP i's dy-

[1]Up to log factors, as shown in Section 4.

[2]Note that in phase 2, once the MDP identity of the present task is known, the knownness threshold can switch to the larger value, without having to wait until all state–actions visitation counts reach the $O(\Gamma^{-2})$ threshold.

namics including reward and transitions: the model dynamics in state–action (s, a) is denoted as an $(S+1)$-dimensional vector $\theta_i(\cdot|s, a)$, where the first S components are the transition probabilities to corresponding next states, and the last component the average reward. The model difference between two MDPs, M_i and M_j, in state–action (s, a) is defined as $\|\theta_i(\cdot|s, a) - \theta_j(\cdot|s, a)\|$, the ℓ_2-difference between their transition probabilities and reward in that state–action. Furthermore, we let N be an upper bound on the number of next states in the transition models in all MDPs in $\mathcal{M}$; while N can be as large as S, it can often be much smaller in many realistic problems.

We make the following assumptions in the analysis:

1. Tasks in $\mathcal{M}$ are drawn from an unknown multinomial distribution, and each task has at least $p_{\min} > 0$ task-prior probability;
2. There is a known upper bound $\bar{C}$ on $C = |\mathcal{M}|$, the number of MDPs in our multi-task RL setting;
3. There is a known gap Γ of model difference in $\mathcal{M}$; that is, for all $M_i, M_j \in \mathcal{M}$, there exists some (s, a) such that $\|\theta_i(\cdot|s, a) - \theta_j(\cdot|s, a)\| > \Gamma$.
4. There is a known diameter D, such that for every MDP in $\mathcal{M}$, any state s' is reachable from any state s in at most D steps *on average*;
5. All tasks are run for $H = \Omega\left(\frac{DSA}{\Gamma^2}\log\frac{T}{\delta}\right)$ steps;

The first assumption essentially ignores extremely rare MDPs. While it is possible to adapt our results to avoid the assumption of $p_{\min}$, we keep it mostly for the sake of simplicity of the exposition. The second assumption says there are not too many different underlying MDPs. In practice, one may choose C to balance flexibility and complexity of multi-task learning. The third assumption says two distinct MDPs in $\mathcal{M}$ must differ by a sufficient amount in their model parameters; otherwise, there would be little need to distinguish them in practice. The fourth assumption about the diameter, introduced by Jaksch et al. [2010], is the major assumption we need in this work. Basically, it ensures that *on average* every state can be reached from other states sufficiently fast. Consequently, it is possible to quickly identify the underlying MDP of a task.

Our main result is the following theorem: the overall sample complexity in solving T tasks is substantially smaller than solving them individually without transfer.

Theorem 1 *Given any ϵ and δ, run Algorithm 1 for T tasks, each for $H = \Omega\left(\frac{DSA}{\Gamma^2}\log\frac{T}{\delta}\right)$ steps. Then, the algorithm will follow an ϵ-optimal policy on all but $\tilde{O}\left(\frac{\zeta V_{\max}}{\epsilon(1-\gamma)}\right)$ steps, with probability at least $1-\delta$, where*

$$\zeta = \tilde{O}\left(T_1\zeta_s + \bar{C}\zeta_s + (T - T_1)\left(\frac{NV_{\max}^2\bar{C}}{\epsilon^2(1-\gamma)^2} + \frac{DC^2}{\Gamma^2}\right)\right),$$

and $\zeta_s = \tilde{O}\left(\frac{NSAV_{\max}^2}{\epsilon^2(1-\gamma)^2}\right)$, *with probability at least* $1-\delta$.

In particular, in phase 2, our Algorithm 1 has a sample complexity that is independent of the size of the state and action spaces, trading this for a dependence on the number of models $\bar{C}$ and diameter D. In contrast, applying single-task learning without transfer in T tasks can lead to an overall sample complexity of $\tilde{O}(T\zeta_s) = \tilde{O}(TNSA)$. Since we expect $\bar{C} \ll SA$, this yields a significant improvement over single-task reinforcement learners (as long as D is not too large), whose sample complexity has at least a linear dependence on the size of the state–action space [Strehl et al., 2006b, Szita and Szepesvári, 2010], and some have a polynomial dependence on the size of the state and/or action spaces. We expect this reduction in sample complexity to also lead to improved empirical performance, and verify this in an experiment later.

A few lemmas are needed to prove the main theorem.

Lemma 1 *If we set $T_1 = p_{\min}^{-1}\ln\bar{C}/\delta$, then with probability $1-\delta$, all MDPs will be encountered in phase 1.*

Proof. In the T_1 samples in phase 1, the probability that every MDP is seen at least once is no smaller than $1 - \bar{C}(1-p_{\min})^{T_1}$. Setting this lower bound to $1-\delta$, solving for T_1, and using the inequality $\ln(1-x) < -x$, we get $T_1 = \frac{1}{p_{\min}}\ln\frac{\bar{C}}{\delta}$ is sufficient. □

Lemma 2 *If all tasks are run for $H = \tilde{O}\left(\frac{DSA}{\Gamma^2}\right)$ steps, then with probability $1-\delta$, the following hold:*

1. *Every state–action in every task receives at least $\Omega(\Gamma^{-2}\ln T/\delta)$ samples from that task;*
2. *The tasks encountered in phase 1 will be grouped correctly with all other tasks corresponding to the same (hidden) MDP;*
3. *Each task in phase 2 will be identified correctly and its counts added to the correct MDP.*

Proof. Assumption 4 ensures that any state is reachable from any other state within $2D$ steps with probability at least 0.5, by Markov's inequality. Chernoff's inequality, combined with a union bound over all T tasks and all SA state–action pairs, implies that with probability at least $1-\delta$, all state–actions can be visited $\Omega(\Gamma^{-2}\ln\frac{TSA}{\delta})$ times as long as sufficiently large H.

The second statement is proved by Hoeffding's inequality. After phase 1 each task will have at least $\Omega(\Gamma^{-2}\ln T/\delta)$ samples for each state–action with high probability. In order to accurately merge tasks into groups implicitly associated with the same underlying MDP, we note that by assumption, any two different MDPs must have dynamics that differ by at least Γ in at least one state–action. In order to detect such a difference, it is sufficient to estimate the models of each state–action to an ℓ_2-accuracy of $\Gamma/4$. In this case, the ℓ_2-difference between any two MDPs must exceed $\Gamma/2$ in at least one state–action pair. A similar analysis of two tasks which come from the same MDP implies that the difference estimated mean rewards can be at most $\Gamma/2$ for all state–actions. This implies that tasks can be clustered into groups corresponding to all tasks from the same MDP by combining tasks whose reward models differ by no more than $\Gamma/2$ across all state–action pairs. This is ensured by Hoeffding's inequality with a union bound, resulting in the sample size of $\Omega\left(\Gamma^{-2}\ln\frac{TSA}{\delta}\right)$.

The third part requires that each tasks's MDP identity can be correctly identified with high probability in phase 2, which can be proved similarly to the second part. □

The next lemma shows, on average, each state transition contains information to distinguish the true MDP model from others. Let θ_1 and θ_2 be two $(S+1)$-dimensional vectors, representing two MDP models for some state–action (s,a). Let $\hat{\theta}_1$ and $\hat{\theta}_2$ be their estimates that have confidence radius $\delta\theta_1$ and $\delta\theta_2$, respectively; that is, $\left\|\theta_1-\hat{\theta}_1\right\|\leq\delta\theta_1$ and similar for θ_2. For a transition (s,a,r,s'), define the square loss of estimated model $\hat{\theta}_i$ by

$$\ell(\hat{\theta}_i)=\sum_{1\leq\tau\leq S,\tau\neq s'}\hat{\theta}_i(\tau)^2+(\hat{\theta}_i(s')-1)^2+(\hat{\theta}_i(S+1)-r)^2.$$

Lemma 3 *If θ_1 is the true model for generating the transition (s,a,r,s'), then*

$$\mathbb{E}_{\theta_1}\left[\ell(\hat{\theta}_2)-\ell(\hat{\theta}_1)\right]\geq\left\|\hat{\theta}_1-\hat{\theta}_2\right\|\left(\left\|\hat{\theta}_1-\hat{\theta}_2\right\|-2\left\|\delta\theta_1\right\|\right).$$

Proof. Written out explicitly, the left-hand side becomes

$$\begin{aligned}&\sum_i\Big(\theta_1(i)\left((1-\hat{\theta}_2(i))^2-(1-\hat{\theta}_1(i))^2\right)\\&\qquad+(1-\theta_1(i))(\hat{\theta}_2(i)^2-\hat{\theta}_1(i)^2)\Big)\\&\qquad+\mathbb{E}_{r\sim\theta_1}\left[(r-\theta_2(S+1))^2-(r-\theta_1(S+1))^2\right].\end{aligned}$$

Some algebra simplifies the above as:

$$\begin{aligned}&\sum_{\tau=1}^{S+1}\left(\hat{\theta}_1(\tau)-\hat{\theta}_2(\tau)\right)\left(2\theta_1(\tau)-\hat{\theta}_1(\tau)-\hat{\theta}_2(\tau)\right)\\&=\sum_{\tau=1}^{S+1}\left(\hat{\theta}_1(\tau)-\hat{\theta}_2(\tau)\right)\left(\hat{\theta}_1(\tau)-\hat{\theta}_2(\tau)2\theta_1(\tau)-2\hat{\theta}_1(\tau)\right)\\&=\left\|\hat{\theta}_1-\hat{\theta}_2\right\|^2+2\langle\hat{\theta}_1-\hat{\theta}_2,\theta_1-\hat{\theta}_1\rangle\\&\geq\left\|\hat{\theta}_1-\hat{\theta}_2\right\|^2-2\left\|\hat{\theta}_1-\hat{\theta}_2\right\|\left\|\theta_1-\hat{\theta}_1\right\|\\&\geq\left\|\hat{\theta}_1-\hat{\theta}_2\right\|\left(\left\|\hat{\theta}_1-\hat{\theta}_2\right\|-2\left\|\delta\theta_1\right\|\right),\end{aligned}$$

where the first inequality is due to Cauchy inequality, and the second to the condition in the lemma. □

We are going to apply a generic PAC-MDP theorem of Strehl et al. [2006a] to analyze the sample complexity of our algorithm. As usual, define a state–action to be known if its reward estimate is within $\Theta(\epsilon(1-\gamma))$ accuracy, and its next-state transition probability estimate is within $\Theta(\epsilon(1-\gamma)/V_{\max})$ in terms of total variation. The next lemma bounds the number of visits to unknown state–actions in the entire second phase.

Lemma 4 *The total number of visits to unknown state–actions in the the second phase is*

$$\tilde{O}\left(\frac{(T-T_1)DC^2}{\Gamma^2}+\frac{NV_{\max}^2C(T-T_1)}{\epsilon^2(1-\gamma)^2}\ln\frac{C}{\delta}+C\zeta_s\right).$$

Proof. (sketch) As explained earlier, Algorithm 2 starts with model identification, and then switches to single-task E^3. The first term in the bound corresponds to the model identification step. Note that, for a set of C models, there are at most C-choose-2, namely $O(C^2)$, many informative states to fully identify a model. Therefore, our algorithm only needs to reach these informative states before figuring out the true model. Similar to the proof of Lemma 2 (part 1), each such state can be visited $\Theta(\Gamma^{-2})$ times in $\tilde{O}(D\Gamma^{-2})$ steps. So, $\tilde{O}(D\Gamma^{-2}C^2)$ steps suffice to visit all these informative states sufficiently often.

The rest of the proof (for the second and third terms) consists of two parts. The first assumes the underlying MDP identity is known at the beginning of each task. In our algorithm, however, the MDP identity is unknown until all but one model is eliminated. Then, the second part shows such a delay of MDP identification is insignificant with respect to the number of visits to unknown state–actions.

We begin with the assumption that the underlying MDP identity is given at the beginning of each task. Although the algorithm knows which MDP it is in in the current task, it still follows the same logic in the pseudocode for model elimination and identification. The only advantage it has is to "boost" its history with samples of previous tasks of the same MDP right after the task begins, rather than at the end of the task. This is like single-task learning by "concatenating" tasks of the same MDP into one big task.

In this scenario, an unknown state–action (s,a) implies at least one of the following must be true. The first is when the number of samples of (s,a) in some model has not exceeded the known threshold $\zeta_s/(SA)$. For this case, since the samples of (s,a) for the same MDP accumulates over tasks, there can be at most $\zeta_s/(SA)$ visits to unknown (s,a) pairs for a single MDP model, and a total of $C\zeta_s/(SA)$ visits to unknown (s,a) across all C models. In the other case, at least two models in $\mathcal{M}$ has a sufficient difference in their estimates of the model parameters for (s,a). Using Lemma 3, one can calculate the expected difference in square loss between the true model and a wrong model. Following similar steps as in [Li et al., 2011],[3] we can see the squared difference on average is at least

[3]The noisy union algorithm of [Li et al., 2011] is based on

$\Theta(\epsilon^2(1-\gamma)^2/(V_{\max}^2 N))$, and after $O(\frac{NV_{\max}^2 C}{\epsilon^2(1-\gamma)^2}\ln\frac{CT}{\delta})$ visits to such state–actions, all models but the true one will be eliminated, with probability at least $1-\frac{\delta}{CT}$. Using a union bound over all tasks in phase 2, we have that, with probability at least $1-\delta$, the same statement holds for all tasks in phase 2. Details will be given in a full version.

The first part of the proof is now completed, showing that when the task identity is given at the beginning of a task, the total number of visits to unknown state–actions is at most $O\left(\frac{NV_{\max}^2 C(T-T_1)}{\epsilon^2(1-\gamma)^2}\ln\frac{C}{\delta}+C\zeta_s\right)$.

We now handle the need for MDP identification, which prevents samples in the present task to contribute to the corresponding model until the underlying MDP is identified. Consider any state–action pair (s,a), and a fixed task in phase 2. At the beginning of the task, the algorithm has access to C models, the i-th of which has accumulated U_i samples for (s,a). After the task, the true MDP (say, model 1, without loss of generality) is identified, whose sample count for (s,a) becomes $U_1' \leftarrow U_1+U_0$, where U_0 is the number of visits to (s,a) in the present task. For other models $i>1$, $U_i' \leftarrow U_i$.

Consider three situations regarding the sample sizes U_1 and U_1'. In the first case, $U_1 < U_1' < \zeta_s/(SA)$, so our multi-task RL algorithm behaves identically no matter whether the samples contribute to the true model estimation immediately or at the end of the task, since (s,a) will remain unknown in either situation. In the second case, $\zeta_s/(SA) \le U_1 < U_1'$, so (s,a) is already know at the beginning of the task, and additional samples for (s,a) does not change the algorithm, or increase the number of visits to unknown state–actions. In the last case, we have $U_1 < \zeta_s/(SA) \le U_1'$. Recall that our algorithm declares a state–action to be known if it has been visited $\zeta_s/(SA)$ times in a single task, so $U_0 \le \zeta_s/(SA)$. Hence, $U_1' = U_1+U_0 < 2\zeta_s/(SA)$. Applying this inequality to all state–actions and all MDPs, we conclude that the number of visits to unknown states is at most $2C\zeta_s$.

Part II above shows the delay in sample accumulation can only cause up to a constant factor increase in the number of visits to unknown state–actions. The lemma follows immediately from the conclusion of part I. □

We are now fully equipped to prove the main result:

Proof. (of Theorem 1) We will use the generic PAC-MDP theorem of Strehl et al. [2006a] by verifying the three needed conditions hold. Although the theorem of [Strehl et al., 2006a] is stated for single-task RL, the proof works without essential changes in multi-task RL.

The first condition holds since the value function is optimistic with high probability, by construction of the known-state MDPs when running E^3 in the tasks.

The second condition also holds. Whenever a state–action becomes known, its reward estimate is within $\epsilon(1-\gamma)$ accuracy, and the transition estimate is within $\epsilon(1-\gamma)/(V_{\max}\sqrt{N})$ accuracy measured by ℓ_2 error. Using the inequality $\|v\|_1 \le \|v\|_2\sqrt{d}$, where d is the dimension of vector v, we know the transition estimate is within $\epsilon(1-\gamma)/V_{\max}$ accuracy measured by total variation. The simulation lemma (see, e.g., [Strehl et al., 2006a]) then implies the accuracy condition holds.

What remains to be shown is that the third condition holds; namely, we will find a bound on the total number of times an unknown state–action pair is visited, across all T tasks. E^3 is executed on each task in phase 1. Prior analysis for Rmax and MBIE (see e.g. [Kakade, 2003, Strehl and Littman, 2008]) applies similarly to E^3, implying that the number of visits to unknown state–action pairs on a single MDP at most $\left(\frac{SANV_{\max}^2}{\epsilon^2(1-\gamma)^2}\right)$. From Lemma 1, after T_1 tasks, all tasks in $\mathcal{C}$ have been encountered with probability at least $1-\delta$. Therefore, with probability at least $1-\delta$, the total number of visits to unknown state–action pairs in phase 1 is at most $\zeta_s T_1$.

In Phase 2, Algorithm 1 runs E^3 in individual tasks but transfers samples from one task to another of the same underlying MDP. We have shown in Lemma 4 the number of visits to unknown state–actions is at most $O\left(\frac{(T-T_1)DC^2}{\Gamma^2}+\frac{NV_{\max}^2 C(T-T_1)}{\epsilon^2(1-\gamma)^2}\ln\frac{C}{\delta}+C\zeta_s\right)$. Hence, the total number of visits to unknown state–actions during all T tasks is at most $O\left(\zeta_s T_1+(T-T_1)\left(\frac{DC^2}{\Gamma^2}+\frac{NV_{\max}^2 C}{\epsilon^2(1-\gamma)^2}\ln\frac{C}{\delta}\right)+C\zeta_s\right)$. The theorem follows immediately by the PAC-MDP theorem of [Strehl et al., 2006a]. □

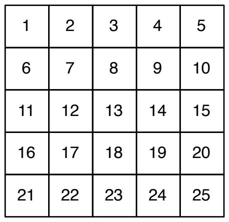


Figure 1: Gridworld domain

5 EXPERIMENTS

Although the main contribution of our paper is to provide the first theoretical justification for online multi-task Rl, we also provide numerical evidence showing the empirical benefit of our proposed approach over single-task learning as well as a state-of-the-art multi-task algorithm.

scalar predictions and observations, while we are dealing with $(S+1)$-dimensional vectors. The only substantial change to their proof is to replace the application of Hoeffding's inequality with its vector-valued extensions, such as Lemma 5 in the appendix.

There are $C = 3$ possible MDPs, each with the same 5×5 state space as shown in the gridworld layout of Figure 5. The start state is always the center state (s_{13}). There are 4 actions that succeed in generally moving the agent in the intended cardinal direction with probability 0.85, going in the other directions (unless there is a wall) with probability 0.05 each. Three of the corners (s_5, s_{21}, s_{25}) exhibit different dynamics: the agent stays in the state with probability 0.95, or otherwise transitions back to the start state. The three MDPs differ only in their reward models. The reward for each state is drawn from a binomial model. Intuitively, in each MDP, one of the corners provides a high reward, two others provide low reward, and there is one additional state whose medium reward can help distinguish the MDPs. More precisely, In MDP 1, s_{21} has a binomial parameter of 0.99, s_6 has a parameter of 0.6, s_5 and s_{25} have a parameter of 0, and all other states have a parameter of 0.1. In MDP 2, s_5 has a binomial parameter of 0.99, s_2 has a parameter of 0.6, s_{21} and s_{25} have a parameter of 0, and all other states have a parameter of 0.1. In MDP 3 s_{25} has a binomial parameter of 0.99, s_1 has a parameter of 0.6, s_{21} and s_{25} have a parameter of 0, and all other states have a parameter of 0.1. A new task is sampled from one of these three MDPs with equal probability.

We compare our proposed approach to the most closely related approach we are aware of, Wilson et al. [2007]'s algorithm on hierarchical multi-task learning (HMTL). Wilson et al. learn a Bayesian mixture model over a set of MDP classes as the agent acts in a series of MDPs, and use prior to transfer knowledge to a new task sampled from one of those classes. When acting in a new MDP, their approach does not explicitly balance exploration and exploitation; instead, it selects the current maximum a posterior (MAP) estimate of the model parameters, computes a policy for this model, and uses this policy for a fixed number of steps, before re-computing the MAP model. Wilson et al. did not provide formal performance guarantees for their approach, but they did achieve promising results on both a simulated domain and a real-time strategy game with HMTL. When applying their approach to our setting, we limit the hierarchy to one level, ensuring that tasks are directly sampled from a mixture of MDPs. We also provide their algorithm with an upper bound on the number of MDPs, though their algorithm is capable to learning this directly.

In both our algorithm and HMTL there are several parameters to be set. For HMTL we set the interval between recomputing the MAP model parameters at 10 steps: this was chosen after informal experimentation suggested this improved performance compared to longer intervals. In our approach we set the threshold for a parameter to be known at $m = 5$. The number of tasks in phase 1 was set to $\lceil 3 \ln(3/0.05) \rceil = 13$, matching the required length specified by Lemma 1. We ran each task for a horizon of $H = 3000$ steps, and performed multi-task reinforcement

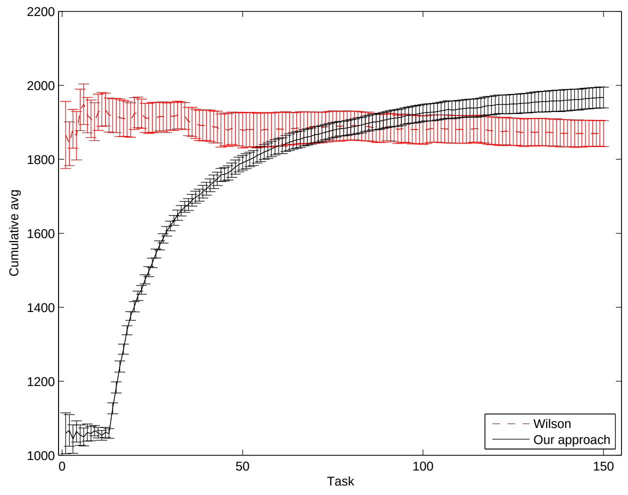


Figure 2: Cumulative average reward per task.

learning across 150 tasks per round. We then repeated this process for 20 rounds.

Figure 5 displays the cumulative per-task reward for each method, averaged across 20 rounds. As expected, our approach performs worse during phase 1, before it has obtained a good estimate of each of the 3 MDPs. In phase 2, our approach performs well, successfully leveraging its knowledge of the models to quickly determine the new task's MDP identity, and then act optimally for that MDP for the remainder of the task. Since phase 1 of our algorithm runs single-task E^3, we can see that transferring knowledge enables our approach to perform substantially better than single-task E^3 ($p < 10^{-4}$ in a Mann-Whitney U test comparing the first task performance to the last).

HMTL does quite well even at the start, because the algorithm directly exploits the current estimated parameters, and once the agent bumps into a good state, the algorithm can leverage that information for the remainder of the task. However, HMTL does not significantly improve beyond its original performance, and performs similarly across the entire length of a multi-task round. We hypothesize that this is because in each task, this approach is not explicitly performing exploration, and therefore may only get good estimates of the parameters of some of the state–action pairs. This means it can be harder to learn a good estimate of the mixture model over the MDPs. Indeed, when we examine the multi-task posterior learned by HMTL, we find that the resulting MDPs appear to be *mixtures* of the true set of MDPs. We compare the total reward obtained in a single round (of all 150 tasks in both phases) of the two approaches, and our approach achieved significantly higher total reward ($p = 0.03$ in a Mann-Whitney U test).

These results provide empirical evidence that our algorithm both achieves significantly better sample-complexity results than prior single-task algorithms as well as a state-of-the-art multi-task algorithm, and these gains can indeed translate to improved empirical performance.

6 RELATED WORK

Our setting most closely matches that of Wilson et al. [2007]; however, they consider a more general two-level hierarchical model where tasks are sampled from a class distribution, and there is a mixture over classes. The authors update a Bayesian prior over the hierarchical model parameters after finishing acting in each task using MCMC, and use and update a local version of this prior during each single-task by sampling an MDP from this prior, following the model's optimal policy for a fixed number of steps, updating the prior, and repeating. Though the authors demonstrate promising empirical learning improvements due to transfer, unlike our work, no formal analysis is provided.

Other work studies the related problem of *transfer RL*. For example, Lazaric and Restelli [2011] provide value-function approximation error bounds of the target task in a *batch* setting, as opposed to our *online* setting where the agent has to balance exploration and exploitation. Their bounds quantify the error potentially introduced by transferring source-task samples to an unrelated target task as well as the reduction in error due to increasing the number of samples from the source. Sorg and Singh [2009] prove bounds on transferring the state–action values from a source MDP to a target MDP, where both MDP models are known and there exists a soft homomorphism between the two state spaces. If the target MDP model is unknown, the authors present a promising heuristic approach without performance guarantees. More recently, Mann and Choe [2012] introduce an algorithm that uses a slight modification of a source task's optimal value as an optimistic initial value for a subset of the target task's state–action pairs, given a mapping between the tasks that ensures the associated value functions have similar values. The authors provide characteristics of this mapping that will improve sample complexity of their algorithm. While interesting, no algorithm was given that could meet these conditions, and no sample-complexity bounds were provided. Perhaps most similar to us is Mehta et al. [2008], who consider sample complexity for transfer learning across semi-MDPs with identical (S, A) and transition models, and a distribution of reward weight vectors. The authors provide a bound on the number of tasks needed until they will be able to immediately identify a close-to-optimal policy for a future task. Compared to our work: (1) the authors only use transferred information to initialize the value function in the new task, (2) their algorithm can produce negative transfer, and more significantly, (3) the authors assume that the model of each new semi-MDP is completely specified.

The model-elimination idea in our Algorithm 2 is related to several previous work on *single-task* RL. The most relevant is probably the (more special) noisy union algorithm of Li et al. [2011] and its application to Met-Rmax [Diuk et al., 2009]. Here, the noisy union algorithm is generalized so that model elimination is possible even before a state–action becomes fully known. Similar ideas are also found in the Parameter Elimination algorithm [Dyagilev et al., 2008], which uses Wald's Sequential Probability Ratio Test (SPRT) to eliminate models, as opposed to the simpler square loss metric we use here. Finally, Lattimore et al. [2013] employ model elimination in their Maximum Exploration algorithm that works in general reinforcement-learning problems beyond MDPs.

7 CONCLUSIONS

In this paper, we analyze the sample complexity of exploration for a multi-task reinforcement-learning algorithm, and show substantial advantage compared to single-task learning. In contrast to the majority of the literature, this work is theoretically grounded, using tools in the PAC-MDP framework. Furthermore, we also show the possibility of avoiding negative transfer in multi-task RL.

These promising results suggest several interesting directions for future research. One of them is to relax some of the assumptions and to develop more broadly applicable algorithms. Second, we intend to test the proposed algorithm in benchmark problems and investigate its empirical advantage compared to single-task RL, as well as its robustness with respect to parameters like $\bar{C}$. Third, it is interesting to extend the current results beyond finite MDPs, possibly relying on function approximation or compact model representations like dynamic Bayes networks [Dean and Kanazawa, 1989]. Finally, our algorithm makes use only of the learned MDP parameters, not of the task distribution over $\mathcal{M}$. Although our own attempt has not yet identified theoretical benefits from such information, we suspect at the least that it will be empirically beneficial.

A A CONCENTRATION INEQUALITY

The following result extends Hoeffding's inequality from real-valued random variables to vector-valued random variables. The tail probability upper bound here is only a constant factor worse than that of Hoeffding's.

Lemma 5 (Hayes [2005]) *For vector-valued martingale,* $\Pr\left(\|X_n\| \geq a\right) \leq 2\exp\left(2 - \frac{a^2}{2n}\right)$*; or equivalently,* $\Pr\left(\left\|\frac{X_n}{n}\right\| \geq \epsilon\right) \leq 2\exp\left(2 - \frac{n\epsilon^2}{2}\right)$.

Acknowledgements

We thank Chris Meek for a helpful discussion of the general differences between transfer learning and multi-task learning. E.B. also thanks Google for partial funding support.

References

R. I. Brafman and M. Tennenholtz. R-max—a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, October 2002.

E. Brunskill. Bayes-optimal reinforcement learning for discrete uncertainty domains. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1385–1386, 2012.

T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

C. Diuk, L. Li, and B. R. Leffler. The adaptive k-meteorologists problem and its application to structure discovery and feature selection in reinforcement learning. In *Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML)*, pages 249–256, 2009.

K. Dyagilev, S. Mannor, and N. Shimkin. Efficient reinforcement learning in parameterized models: Discrete parameter case. In *Recent Advances in Reinforcement Learning*, volume 5323 of *Lecture Notes in Computer Science*, pages 41–54, 2008.

T. P. Hayes. A large-deviation inequality for vector-valued martingales, 2005. Unpublished manuscript.

T. Jaksch, R. Ortner, and P. Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563–1600, 2010.

S. M. Kakade. *On the Sample Complexity of Reinforcement Learning.* . PhD thesis, University College London, 2003.

M. J. Kearns and S. P. Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2–3):209–232, 2002.

T. Lattimore, M. Hutter, and P. Sunehag. The sample-complexity of general reinforcement learning. In *Proceedings of Thirtieth International Conference on Machine Learning (ICML)*, 2013. To appear.

A. Lazaric and M. Restelli. Transfer from multiple MDPs. In *Proceedings of the Neural Information Processing Systems (NIPS)*, pages 1746–1754, 2011.

L. Li, M. L. Littman, T. J. Walsh, and A. L. Strehl. Knows what it knows: A framework for self-aware learning. *Machine Learning*, 82(3):399–443, 2011.

T. A. Mann and Y. Choe. Directed exploration in reinforcement learning with transferred knowledge. In *European Workshop on Reinforcement Learning*, 2012.

N. Mehta, S. Natarajan, P. Tadepalli, and A. Fern. Transfer in variable-reward hierarchical reinforcement learning. *Machine Learning*, 73(3):289–312, 2008.

P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete Bayesian reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 697–704, 2006.

M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, New York, 1994. ISBN 0-471-61977-9.

J. Sorg and S. P. Singh. Transfer via soft homomorphisms. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent System (AAMAS)*, pages 741–748, 2009.

A. L. Strehl and M. L. Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

A. L. Strehl, L. Li, and M. L. Littman. Incremental model-based learners with formal learning-time guarantees. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 485–493, 2006a.

A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman. PAC model-free reinforcement learning. In *Proceedings of the Twenty-Third International Conference on Machine Learning (ICML)*, pages 881–888, 2006b.

A. L. Strehl, L. Li, and M. L. Littman. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10:2413–2444, 2009.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA, March 1998. ISBN 0-262-19398-1.

I. Szita and C. Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning (ICML)*, pages 1031–1038, 2010.

M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(1):1633–1685, 2009.

A. Wilson, A. Fern, S. Ray, and P. Tadepalli. Multi-task reinforcement learning: a hierarchical Bayesian approach. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1015–1022, 2007.

Automorphism Groups of Graphical Models and Lifted Variational Inference

Hung Hai Bui
Natural Language Understanding Lab
Nuance Communications
bui.h.hung@gmail.com

Tuyen N. Huynh
Artificial Intelligence Center
SRI International
huynh@ai.sri.com

Sebastian Riedel
Department of Computer Science
University College London
sebastian.riedel@gmail.com

Abstract

Using the theory of group action, we first introduce the concept of the *automorphism group* of an exponential family or a graphical model, thus formalizing the general notion of symmetry of a probabilistic model. This automorphism group provides a precise mathematical framework for lifted inference in the general exponential family. Its group action partitions the set of random variables and feature functions into equivalent classes (called orbits) having identical marginals and expectations. Then the inference problem is effectively reduced to that of computing marginals or expectations for each class, thus avoiding the need to deal with each individual variable or feature. We demonstrate the usefulness of this general framework in lifting two classes of variational approximation for maximum a posteriori (MAP) inference: local linear programming (LP) relaxation and local LP relaxation with cycle constraints; the latter yields the first lifted variational inference algorithm that operates on a bound tighter than the local constraints.

1 Introduction

Classical approaches to probabilistic inference—an area now reasonably well understood—have traditionally exploited low tree-width and sparsity of the graphical model for efficient exact and approximate inference. A more recent approach known as *lifted inference* [4, 16, 7, 8] has demonstrated the possibility to perform very efficient inference in highly-connected, but *symmetric* models, such as those arising in the context of relational (first-order) probabilistic models.

Symmetry is the essential element of lifted inference. But currently, no formally defined notion of symmetry of a probabilistic model exists, and thus no formal account of what "exploiting symmetry" means in lifted inference has been defined. As a result, most previous work has derived lifted versions of existing propositional algorithms from a *procedural* perspective: for models that exhibit symmetries, propositional inference algorithms tend to perform the same computations several times, and their lifted counterparts are designed to perform these operation once. This approach severely limits the theoretical understanding of the nature of lifted inference. In practice, this approach also limits the class of inference algorithms that we can lift. For example, many ground inference updates (e.g., asynchronous belief propagation, max-product linear programming (MPLP) [5]) are made in a sequence that breaks the symmetry of the original model. Likewise, with the advance in modern optimization, many algorithms rely on off-the-shelf solvers in their inner loop, and lifting these solvers is not practical.

In this work, we propose an alternative approach: rather than lifting inference *algorithms*, we lift their *variational formulations*, the optimization problems that variational inference algorithms seek to solve. These lifted formulations can then be tackled with the usual optimization toolbox (off-the-shelf solvers, cutting plane algorithms, dual block coordinate descent updates etc.). If the original model exhibits symmetry, then the lifted formulations will generally be more compact than their propositional counterparts, and hence their optimization is likely to be more efficient. This *declarative* approach to lifting gives rise to a new class of algorithms, including the first lifted variational algorithm that operates on a bound tighter than the local constraints.

This paper is divided into three parts: In the first part, we show how to find a *lifting partition*: sets of random variables and feature functions that have identical expectations. We present a formal account of symmetry in graphical models through automorphism groups of exponential families. When there is parameter-tying, the automorphism group leads to a subgroup, termed the *lifting group*, which also captures symmetry in the parameters. By linking the lifting group to the well-known subject of *graph automorphisms* [10, 6], we can leverage off-the-shelf tools to find lifting partitions as orbits of the lifting group. Further, by connecting the lifting group to *renaming* permutations of logical constants in Markov Logic Network (MLN) [14], we find lifting partitions without unrolling the MLN. In work done concurrently and independently from ours, Niepert [12, 13] presented similar ideas for exploiting orbits of permutation groups in lifting Markov Chain Monte Carlo (MCMC) algorithms. Though the ideas are similar, unique to our contribution is the rigorously defined automorphism group of a general exponential family that enables formal proofs of all subsequent results.

In the second part, we are given a lifting partition, and we use it to collapse the variational variables and constraint set. In particular, we investigate two popular variational relaxations of MAP inference. The first one is based on the local polytope, and the second one is based on a tightening of the local polytope with cycle constraints. For the latter, we also develop a lifted separation oracle to find violated constraints in the reduced yet still exponential lifted cycle polytope.

In the third part, we evaluate the novel algorithms that our framework gives rise to. Using an off-the-shelf LP solver, we show that for models with symmetry, lifted MAP in the local polytope is more efficient than propositional MAP. Likewise, for models with symmetry and repulsion, the lifted cycle polytope yields more accurate results than its local counterpart, and requires less runtime than the propositional version. Finally, we show the effectiveness of the renaming approach to finding lifting partitions. Although the proofs are non-trivial, due to space restrictions, they are omitted but can be found in [3].

2 Background on Groups and Graph Automorphisms

A *partition* $\Delta = \{\Delta_1 \ldots \Delta_k\}$ of a set V is a set of disjoint nonempty subsets of V whose union is V. Each element Δ_i is called a *cell*; $|\Delta|$ is thus the number of cells or the *size* of the partition. A partition Δ defines an equivalence relation $\overset{\Delta}{\sim}$ on V by letting $u \overset{\Delta}{\sim} v$ iff u and v are in the same cell. A partition Λ is finer than Δ if every cell of Λ is a subset of some cell of Δ.

We now briefly review the important concepts in group theory and graph automorphisms [6]. A mathematical *group* $(\mathbb{G}, \cdot)$ is a non-empty set $\mathbb{G}$ containing an identity element, denoted by $\mathbf{1}$, and a binary operation $\cdot$ which is associative and closed in $\mathbb{G}$. The group identity satisfies $\forall g \in \mathbb{G}$, $\mathbf{1} \cdot g = g \cdot \mathbf{1} = g$, and every element of $\mathbb{G}$ is invertible, i.e., $\exists g^{-1}$ such that $g \cdot g^{-1} = g^{-1} \cdot g = \mathbf{1}$. A group containing $\mathbf{1}$ as its only element is called a trivial *group. A subgroup* of $\mathbb{G}$ is a subset of $\mathbb{G}$ that forms a group with the same binary operation as $\mathbb{G}$. We write $\mathbb{G}_1 \leq \mathbb{G}_2$ when $\mathbb{G}_1$ is a subgroup[1] of $\mathbb{G}_2$.

A permutation of a set V is a bijective mapping from V to itself. Two permutations can be composed together via the usual composition of two mappings. Any set of permutations (on V) that contains the identity permutation and is closed under composition and taking inverse thus forms a group. The set of *all* permutations of V is called the *symmetric group* $\mathbb{S}(V)$. The symmetric group $\mathbb{S}_n$ is the set of all permutations of $\{1, 2, \ldots, n\}$. For a permutation $\pi \in \mathbb{S}_n$, $\pi(i)$ is the image of i under π. For each vector $x \in \mathcal{X}^n$, the vector x permuted by π, denoted by x^π, is $(x_{\pi(1)} \ldots x_{\pi(n)})$; for a set $A \subset \mathcal{X}^n$, the set A permuted by π, denoted by A^π is $\{x^\pi | x \in A\}$.

A subgroup $\mathbb{G}$ of $\mathbb{S}(V)$ induces the following equivalence relation on V: $v \sim v'$ iff there exists $g \in \mathbb{G}$ such that $g(v) = v'$ (the fact that $\sim$ is an equivalence relation follows from the definition of a group). $\mathbb{G}$ therefore induces a partition on V, called the *orbit partition,* denoted by $\mathrm{Orb}_{\mathbb{G}}(V)$. The *orbit* of an element $v \in V$ is the set of elements in V equivalent to v: $\mathrm{orb}_{\mathbb{G}}(v) = \{v' \in \mathcal{V} | \, v' \sim v\}$.

A group $\mathbb{G}$ can induce an orbit partition on any set U as long as members of $\mathbb{G}$ can be viewed as (not necessarily distinct) permutations of U. In this case, there is a group homomorphism from $\mathbb{G}$ to a subgroup of $\mathbb{S}(U)$, and the group $\mathbb{G}$ is said to *act* on the set U. A subgroup $\mathbb{G}_1 \leq \mathbb{G}$ will also act on U and induces a finer orbit partition. Given a set element $u \in U$ and a group element $g \in \mathbb{G}$, if $g(u) = u$ then g is said to stabilize u. If $\forall g \in \mathbb{G}$, $g(u) = u$, then the group $\mathbb{G}$ is said to stabilize u.

Group action is a powerful concept since it allows the same group $\mathbb{G}$ to act (hence induce orbit partitions) on many different sets. For example, $\mathbb{S}_n$ acts on the set of n-dimension vectors $\mathcal{X}^n$ via the action $\pi(x) = x^\pi$. $\mathbb{S}_n$ also acts on the set of n-vertex graphs in the following way. Every permutation $\pi \in \mathbb{S}_n$ transforms a graph $\mathfrak{G}$ to its isomorphic variant $\mathfrak{G}'$ (i.e., $\{i, j\}$ is an edge in $\mathfrak{G}$ iff $\{\pi(i), \pi(j)\}$ is an edge in $\mathfrak{G}'$). Hence, it can be viewed as a bijection (permutation) on the set of n-vertex graphs. If $\pi(\mathfrak{G}) = \mathfrak{G}$ then π stabilizes $\mathfrak{G}$ and is called an *automorphism* of the graph $\mathfrak{G}$. The set of all automorphisms of $\mathfrak{G}$ forms a group named the *automorphism group* of $\mathfrak{G}$, denoted by $\mathbb{A}(\mathfrak{G})$ (see Figure 1). It is clear that $\mathbb{A}(\mathfrak{G})$ is a subgroup of $\mathbb{S}_n$. The cardinality of $\mathbb{A}(\mathfrak{G})$ indicates the level of symmetry in $\mathfrak{G}$. If $\mathbb{A}(\mathfrak{G})$ is the trivial group then $\mathfrak{G}$ is asymmetric; if $\mathbb{A}(\mathfrak{G}) = \mathbb{S}_n$ then $\mathfrak{G}$ either is fully connected or has no edges. This concept of graph automorphism directly generalizes to graphs with additional structures such as directions, colors, etc.

If we now ask what elements of $\mathfrak{G}$ are indistinguishable up to symmetry, the automorphism group $\mathbb{A}(\mathfrak{G})$ can give us the precise answer. For example, if v' can be obtained from a node v via some permutation π in $\mathbb{A}(\mathfrak{G})$, then these two nodes are indistinguishable and must have the same the graph properties (e.g., degree, averaged distance to other nodes, etc.). $\mathbb{A}(\mathfrak{G})$ thus partitions the set of nodes V into the node-orbits $\mathrm{Orb}_{\mathbb{A}(\mathfrak{G})}(V)$ where each node orbit is a set of vertices equivalent to one another up to some node relabeling. Furthermore, $\mathbb{A}(\mathfrak{G})$ also acts on the set of graph edges E of $\mathfrak{G}$ by letting $\pi(\{u, v\}) = \{\pi(u), \pi(v)\}$ and this action partitions E into a set of edge-orbits $\mathrm{Orb}_{\mathbb{A}(\mathfrak{G})}(E)$. Similarly, we can also obtain the set of arc-orbits $\mathrm{Orb}_{\mathbb{A}(\mathfrak{G})}(\overrightarrow{E})$.

Computing the automorphism group of a graph is as difficult as determining whether two graphs are isomorphic, a

[1]We use the notation $\mathbb{G}_1 \preceq \mathbb{G}_2$ to mean $\mathbb{G}_1$ is isomorphic to a subgroup of $\mathbb{G}_2$.

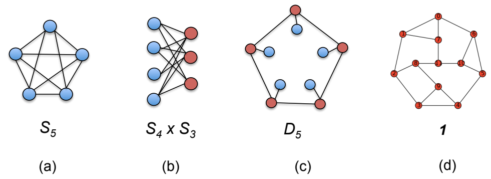


Figure 1: Graphs and their automorphism groups: (a) $\mathbb{A}(K_5) = \mathbb{S}_5$; (b) $\mathbb{A}(K_{4\times3}) = \mathbb{S}_4 \times \mathbb{S}_3$; (c) this graph can be rotated or flipped, yielding the automorphism dihedral group D_5; and (d) this is known as the Frucht's graph, a regular but asymmetric graph. Blue and red colors in (a)-(c) denote different node orbits.

problem that is known to be in NP, but for which it is unknown whether it has a polynomial time algorithm or is NP-complete. In practice, efficient computer programs, such as *nauty*[2] [10], exist for computing automorphism groups of graphs.

3 Symmetry of the Exponential Family

3.1 Exponential Family and Graphical Model

Consider an exponential family over n random variables $(x_i)_{i\in\mathcal{V}}$ where $\mathcal{V} = \{1\ldots n\}$, $x_i \in \mathcal{X}$ with density function

$$\mathcal{F}(x \mid \theta) = h(x)\exp\left(\langle\Phi(x),\theta\rangle - A(\theta)\right)$$

where h is the base density, $\Phi(x) = (\phi_j(x))_{j\in\mathcal{I}}$, $\mathcal{I} = \{1,2,\ldots,m\}$ is an m-dimensional feature vector, $\theta \in \mathbb{R}^m$ is the natural parameter, and $A(\theta)$ the log-partition function. Let $\Theta = \{\theta \,|A(\theta) < \infty\}$ be the set of natural parameters, $\mathcal{M} = \{\mu \in \mathbb{R}^m \mid \exists p,\ \mu = \mathrm{E}_p\Phi(x)\}$ the set of realizable mean parameters, $A^* : \mathcal{M} \to \mathbb{R}$ the convex dual of A, and $\mathbf{m} : \Theta \to \mathcal{M}$ the mean parameter mapping that maps $\theta \mapsto \mathbf{m}(\theta) = \mathrm{E}_\theta\Phi(x)$. Note that $\mathbf{m}(\Theta) = \mathrm{ri}\,\mathcal{M}$ is the relative interior of $\mathcal{M}$. For more details, see [19].

Often, a feature function ϕ_i depends only on a subset of the variables in $\mathcal{V}$. In this case we will write ϕ_i more compactly in factorized form as $\phi_i(x) = \mathrm{f}_i(x_{i_1}\ldots x_{i_K})$ where the indices i_j are distinct, $i_1 < i_2 \ldots < i_K$, and f_i cannot be reduced further, i.e., it must depend on all of its arguments. To keep track of variable indices of arguments of f_i, we let $scope(\mathrm{f}_i)$ denote its set of arguments, $\eta_i(k) = i_k$ the k-th argument and $|\eta_i|$ its number of arguments. Factored forms of features can be encoded as a hypergraph $\mathcal{G}\,[\mathcal{F}]$ of $\mathcal{F}$ (called the graph structure or graphical model of $\mathcal{F}$) with nodes $\mathcal{V}$, and hyperedges (clusters) $\{C|\exists i, scope(\mathrm{f}_i) = C\}$. For models with pairwise features, $\mathcal{G}$ is a standard graph.

For discrete random variables (i.e., $\mathcal{X}$ is finite), we often want to work with the overcomplete family $\mathcal{F}^o$ that we now describe for the case with pairwise features. The set of overcomplete features $\mathcal{I}^o$ are indicator functions on the nodes and edges of the graphical model $\mathcal{G}$ of $\mathcal{F}$: $\phi^o_{u:t}(x) = \mathbb{I}\{x_u = t\}, t \in \mathcal{X}$ for each node $u \in V(\mathcal{G})$; and $\phi^o_{\{u:t,v:t'\}}(x) = \mathbb{I}\{x_u = t, x_v = t'\}, t,t' \in \mathcal{X}$ for each edge $\{u,v\} \in E(\mathcal{G})$. The set of overcomplete realizable mean parameters $\mathcal{M}^o$ is also called the *marginal polytope* because the overcomplete mean parameter corresponds to node and edge marginal probabilities. Given a parameter θ, the transformation of $\mathcal{F}(x|\theta)$ to its overcomplete representation is done by letting θ^o be the corresponding parameter in the overcomplete family: $\theta^o_{u:t} = \sum_{i \text{ s.t. } scope(\mathrm{f}_i)=\{u\}} \mathrm{f}_i(t)\theta_i$ and (assuming $u < v$) $\theta^o_{\{u:t,v:t'\}} = \sum_{i \text{ s.t. } scope(\mathrm{f}_i)=\{u,v\}} \mathrm{f}_i(t,t')\theta_i$. Verifying that $\mathcal{F}^o(x|\theta^o) = \mathcal{F}(x|\theta)$ is straightforward.

3.2 Automorphism Group of an Exponential Family

We define the symmetry of an exponential family $\mathcal{F}$ as the group of transformations that preserve $\mathcal{F}$ (hence preserve h and Φ). The kind of transformation used will be a pair of permutations (π,γ) where π permutes the set of variables and γ permutes the feature vector.

Definition 1. An automorphism of the exponential family F is a pair of permutations (π,γ) where $\pi \in \mathbb{S}_n$, $\gamma \in \mathbb{S}_m$ such that for all vectors x: $h(x^\pi) = h(x)$ and $\Phi^{\gamma^{-1}}(x^\pi) = \Phi(x)$ (or equivalently, $\Phi(x^\pi) = \Phi^\gamma(x)$).

Showing that the set of all automorphisms of $\mathcal{F}$, denoted by $\mathbb{A}[\mathcal{F}]$, forms a subgroup of $\mathbb{S}_n \times \mathbb{S}_m$ is straightforward. This group acts on $\mathcal{I}$ by the permuting action of γ, and on $\mathcal{V}$ by the permuting action of π. In the remainder of this paper, h is always a symmetric function (e.g., $h \equiv 1$); therefore, the condition $h(x^\pi) = h(x)$ automatically holds.

Example 1. Let $\mathcal{V} = \{1\ldots4\}$ and $\Phi = \{\mathrm{f}_1\ldots\mathrm{f}_5\}$ where $\mathrm{f}_1(x_1,x_2) = x_1(1-x_2)$, $\mathrm{f}_2(x_1,x_3) = x_1(1-x_3)$, $\mathrm{f}_3(x_2,x_3) = x_2x_3$, $\mathrm{f}_4(x_2,x_4) = x_4(1-x_2)$, $\mathrm{f}_5(x_3,x_4) = x_4(1-x_3)$. Then $\pi = (1 \leftrightarrow 4)\,(2 \leftrightarrow 3)$, $\gamma = (1 \leftrightarrow 5)\,(2 \leftrightarrow 4)$ form an automorphism of $\mathcal{F}$, since $\Phi^{\gamma^{-1}}(x^\pi) = (\phi_5(x_4\ldots x_1), \phi_4(x_4\ldots x_1), \ldots, \phi_1(x_4\ldots x_1)) = (\mathrm{f}_5(x_2,x_1), \mathrm{f}_4(x_3,x_1), \mathrm{f}_3(x_3,x_2), \mathrm{f}_2(x_4,x_2), \mathrm{f}_1(x_4,x_3)) = (x_1(1-x_2), x_1(1-x_3), x_3x_2, x_4(1-x_2), x_4(1-x_3)) = \Phi(x_1\ldots x_4)$.

An automorphism as defined above preserves a number of key characteristics of the exponential family $\mathcal{F}$ (such as its natural parameter space, its mean parameter space, and its log-partition function), as shown in the following theorem.

Theorem 1. *If* $(\pi,\gamma) \in \mathbb{A}[\mathcal{F}]$ *then*

1. $\pi \in \mathbb{A}(\mathcal{G}[\mathcal{F}])$, i.e. π is an automorphism of the graphical model graph $\mathcal{G}[\mathcal{F}]$.
2. $\Theta^\gamma = \Theta$ and $A(\theta^\gamma) = A(\theta)$ for all $\theta \in \Theta$.
3. $\mathcal{F}(x^\pi|\theta^\gamma) = \mathcal{F}(x|\theta)$ for all $x \in \mathcal{X}^n$, $\theta \in \Theta$.
4. $\mathbf{m}^\gamma(\theta) = \mathbf{m}(\theta^\gamma)$ for all $\theta \in \Theta$.
5. $\mathcal{M}^\gamma = \mathcal{M}$ and $A^*(\mu^\gamma) = A^*(\mu)$ for all $\mu \in \mathcal{M}$.

3.3 Parameter Tying and the Lifting Group

We now consider a parameter-tying setting where some components of θ are the same. Formally, a partition Δ of $\mathcal{I}$ is called the *parameter-tying partition* iff $j \overset{\Delta}{\sim} j' \Rightarrow \theta_j = \theta_{j'}$. Let $\mathbb{R}^m_\Delta$ denote the subspace $\left\{r \in \mathbb{R}^m \mid r_j = r_{j'} \text{ if } j \overset{\Delta}{\sim} j'\right\}$. For any set $S \subset \mathbb{R}^m$, let

[2]*http://cs.anu.edu.au/people/bdm/nauty/*

S_Δ denote the set intersection $S \cap \mathbb{R}^m_\Delta$. Parameter tying is equivalent to restricting the natural parameter θ to the set Θ_Δ. This is also equivalent to working with a different exponential family with $|\Delta|$ aggregating features $\left(\sum_{j\in\Delta_i}\phi_j\right)_i$. While this family has fewer parameters, it is not obvious how it would help inference; moreover, in working directly with the aggregation features, the structure of the original family is lost. Our goal is to study how parameter-tying, coupled with the symmetry of the family $\mathcal{F}$, can lead to more efficient inference.

The automorphism group $\mathbb{A}[\mathcal{F}]$ preserves the family of distributions $\mathcal{F}$; however, this group does not take any specific parameter θ into account. Of special interest is the set of automorphisms that also preserve θ for every tied parameter $\theta \in \Theta_\Delta$. We will now formalize this concept. Given a partition Δ, a permutation λ on $\mathcal{I}$ is consistent with Δ iff λ permutes only among elements of the same cell of Δ. Clearly, for all $\theta \in \Theta_\Delta$, $\theta^\lambda = \theta$. If $\mathbb{G}$ is a group acting on $\mathcal{I}$, we let $\mathbb{G}_\Delta$ denote the set of group elements whose actions are consistent with Δ, that is $\mathbb{G}_\Delta = \left\{g \in \mathbb{G} | \forall u \in \mathcal{I},\ g(u) \overset{\Delta}{\sim} u\right\}$. It is straightforward to verify that $\mathbb{G}_\Delta$ is a subgroup of $\mathbb{G}$.

Definition 2. (Lifting Group) The lifting group corresponding to the parameter-tying partition Δ is $\mathbb{A}_\Delta(\mathcal{F})$, the subgroup of $\mathbb{A}[\mathcal{F}]$ whose member's action is consistent with Δ.

The lifting group $\mathbb{A}_\Delta(\mathcal{F})$ thus stabilizes not just the family $\mathcal{F}$, but also every parameter $\theta \in \Theta_\Delta$. Furthermore, features in the same orbit induced by the lifting group must have the same expectation (a consequence of theorem 1, part 4). As we shall see in the later section, the lifting group $\mathbb{A}_\Delta(\mathcal{F})$ and its induced orbit partitions on the set of variables and features play a central role in our lifted variational inference framework.

4 Detecting Symmetries in Exponential Families

We now discuss the computation of the lifting group $\mathbb{A}_\Delta(\mathcal{F})$ and its orbit partitions. In practice, computing and working with a subgroup of the lifting group suffice.

4.1 Detecting Symmetries via Graph Automorphisms

Our first approach is to construct a suitable graph whose automorphism group is guaranteed to be a subgroup of $\mathbb{A}_\Delta(\mathcal{F})$, and thus any tool and algorithm for computing graph automorphism can be applied. The constructed graph resembles a factor graph representation of $\mathcal{F}$. However, we also use colors of factor nodes to mark feature functions that are both identical and in the same cell of Δ, and colors of edges to encode symmetry of the feature functions themselves.

Definition 3. The colored factor graph induced by $\mathcal{F}$ and Δ, denoted by $\mathfrak{G}_\Delta[\mathcal{F}]$ is a bipartite graph with nodes $V(\mathfrak{G}) = \{x_1 \dots x_n\} \cup \{\mathrm{f}_i \dots \mathrm{f}_m\}$ and edges $E(\mathfrak{G}) = \left\{\{x_{\eta_i(k)}, f_i\} \mid i \in \mathcal{I},\ k = 1 \dots |\eta_i|\right\}$. Variable nodes are assigned the same color which is different from the colors of factor nodes. Factor nodes f_i and f_j have the same color iff $\mathrm{f}_i \equiv \mathrm{f}_j$ and $i \overset{\Delta}{\sim} j$. If the function f_i is symmetric, then all edges adjacent to f_i have the same color; otherwise, they are colored according to the argument number of f_i, i.e., $\{x_{\eta_i(k)}, \mathrm{f}_i\}$ is assigned the k-th color.

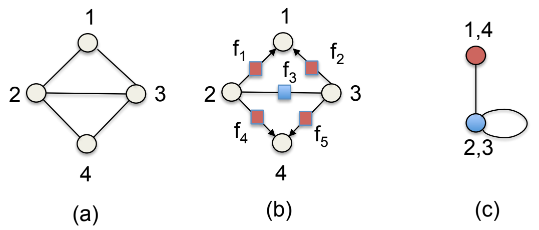


Figure 2: Graph construction for computing the lifting group and its orbits: (a) original graphical model of example 1; (b) constructed colored factor graphs, assuming all parameters are the same (arrows represent first arguments of the asymmetric factors); and (c) lifted graphical model with nodes representing node orbits and edges representing edge orbits of the original graphical model.

Figure 2 shows the construction of the colored factor graph for the exponential family in example 1 where we have assumed that all the parameters are the same.

Theorem 2. *The automorphism group* $\mathbb{A}[\mathfrak{G}_\Delta]$ of $\mathfrak{G}_\Delta[\mathcal{F}]$ *is a subgroup of* $\mathbb{A}_\Delta(\mathcal{F})$*, i.e.,* $\mathbb{A}[\mathfrak{G}_\Delta] \le \mathbb{A}_\Delta[\mathcal{F}]$.

Finding the automorphism group $\mathbb{A}[\mathfrak{G}_\Delta]$ of the graph $\mathfrak{G}_\Delta[\mathcal{F}]$ therefore yields a procedure to compute a subgroup of $\mathbb{A}_\Delta[\mathcal{F}]$. *Nauty*, for example, directly implements operations of computing the automorphism group of a graph and extracting the induced node orbits and edge orbits.

4.2 Symmetries of Markov Logic Networks

Markov Logic Network (MLN) [14] is a first-order probabilistic model that defines an exponential family on random structures (i.e., random graphs, hypergraphs, or more generally random Herbrand models of the first-order language). In this case, a subgroup of the lifting group can be obtained via the symmetry of the unobserved constants in the domain without the need to consider the ground graphical model.

An MLN is prescribed by a list of weighted formulas $F_1 \dots F_K$ (consisting of a set of predicates, logical variables, constants, and a weight vector $\mathbf{w}$) and a logical domain $\mathcal{D} = \{a_1 ... a_{|\mathcal{D}|}\}$. Let $\mathcal{D}_0$ be the set of objects appearing as constants in these formulas, then $\mathcal{D}_* = \mathcal{D}\backslash\mathcal{D}_0$ is the set of objects in $\mathcal{D}$ that do not appear in these formulas. Let Gr be the set of all ground predicates $p(a_1 \dots a_\ell)$'s. Given a substitution s, $F_i[s]$ denotes the result of applying the substitution s to F_i and is a grounding of F_i if it does not contain any free logical variables. The set of all groundings of F_i is GrF_i, and let $\mathrm{GrF} = \mathrm{GrF}_1 \cup \dots \cup \mathrm{GrF}_K$. Let ω be a truth assignment to all the ground predicates in Gr and w_i be the weight of the formula F_i. The MLN corresponds to an exponential family $\mathcal{F}_{MLN}$ where Gr is the variable index set and each grounding $F_i[s] \in \mathrm{GrF}_i$ is a feature function $\phi_{F_i[s]}(\omega) = \mathbb{I}(\omega \vDash F_i[s])$ with the associated parameter $\theta_{F_i[s]} = \mathsf{w}_i$. Since all the ground features of the formula F_i have the same parameter w_i, the MLN also induces the parameter-tying partition $\Delta_{MLN} = \{\{\phi_{F_1[s]}(\omega)\} \dots \{\phi_{F_K[s]}(\omega)\}\}$.

Let a renaming permutation r be a permutation over $\mathcal{D}$ that fixes every object in $\mathcal{D}_0$ (i.e., r only permutes objects in $\mathcal{D}_*$). Thus, the set of all such renaming permutations is a group $\mathbb{G}^{re}$ isomorphic to the symmetric group $\mathbb{S}(\mathcal{D}_*)$. Consider the following action of $\mathbb{G}^{re}$ on Gr : $\pi_r : p(a_1 \dots a_\ell) \mapsto p(r(a_1) \dots r(a_\ell))$, and the action on GrF $\gamma_r : F_i[s] \mapsto F_i[r(s)]$ where $r(s = (x_1/a_1, ..., x_k/a_k)) = (x_1/r(a_1), ..., x_k/r(a_k))$. Intuitively, π_r and γ_r rename the constants in each ground predicate $p(a_1 \dots a_\ell)$ and ground formula $F_i[s]$ according to the renaming permutation r. The following is a consequence of Lemma 1 from Bui et al. [2].

Theorem 3. *For every renaming permutation r, $(\pi_r, \gamma_r) \in \mathbb{A}[\mathcal{F}_{MLN}]$. Further, the renaming group $\mathbb{G}^{re}$ is isomorphic to a subgroup of the MLN's lifting group: $\mathbb{G}^{re} \preceq \mathbb{A}_{\Delta_{MLN}}[\mathcal{F}_{MLN}]$.*

Orbit partitions induced by $\mathbb{G}^{re}$ on the set of predicate groundings can be derived directly from the first-order representation of an MLN without considering its ground graphical model. The size of this orbit partition depends only on the number of observed constants $|\mathcal{D}_o|$, and does not depend on actual domain size $|\mathcal{D}|$. For example, if $q(.,.)$ is a 2-ary predicate and there is one observed constant a, then we obtain the following partition of the groundings of q: $\{q(a,a)\}$, $\{q(x,x)|x \neq a\}$, $\{q(a,x)|x \neq a\}$, $\{q(x,a)|x \neq a\}$, $\{q(x,y)|x \neq y, x \neq a, y \neq a\}$. Similar partitions on the set of factors and variable clusters can also be obtained with complexity polynomial in $|\mathcal{D}_o|$ and independent of $|\mathcal{D}|$.

5 Lifted Variational Inference Framework

We now discuss the principle of how to exploit the symmetry of the exponential family graphical model for lifted variational inference. In the general variational inference framework [19], marginal inference is viewed as a means to compute the mean parameter $\mu = \mathbf{m}(\theta)$ given a natural parameter θ by solving the optimization problem

$$\sup_{\mu \in \mathcal{M}} \langle \theta, \mu \rangle - A^*(\mu). \tag{1}$$

For discrete models, the variational problem is more conveniently posed using the overcomplete parameterization, for marginal and MAP inference

$$\sup_{\mu^o \in \mathcal{M}^o} \langle \mu^o, \theta^o \rangle - A^{o*}(\mu^o) \tag{2}$$

$$\max_{x \in \mathcal{X}^n} \ln \mathcal{F}(x|\theta) = \sup_{\mu^o \in \mathcal{M}^o} \langle \mu^o, \theta^o \rangle + \text{const.} \tag{3}$$

We first focus on lifting the main variational problem in (1) and leave discussions of the other problems to subsection 5.3.

5.1 Lifting Partition

Consider the parameter-tying scenario where $\theta \in \Theta_\Delta$ for a given partition Δ on the feature set $\mathcal{I}$. With this restriction, the mean parameter by definition must lie inside $\mathbf{m}(\Theta_\Delta)$,

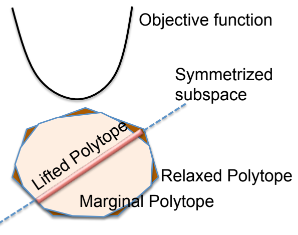


Figure 3: (Best viewed in color) Symmetrized subspace

so in theory, the domain of the variational optimization problems can be restricted to $\mathbf{m}(\Theta_\Delta)$. The main difficulty here lies in how to characterize $\mathbf{m}(\Theta_\Delta)$.

We first make a rather intuitive observation: for general convex optimization problems with symmetric objective functions and constraints, the optimal solutions are trapped in a lower-dimensional symmetrized subspace (see Figure 5.1). This is formalized in lemma 1, whose proof makes use of the orbit-stabilizer theorem, an elementary result in group theory.

Definition 4. (Lifting partition) Consider the convex optimization $\inf_{\mathbf{x} \in \mathcal{S}} J(\mathbf{x})$ where $\mathcal{S} \subset \mathbb{R}^m$ is a convex set and J is a convex function. A partition φ of $\{1 \dots m\}$ is a *lifting partition* for the aforementioned problem iff $\inf_{x \in S} J(x) = \inf_{x \in S_\varphi} J(x)$ (i.e., the constraint set S can be restricted to $S_\varphi = S \cap \mathbb{R}^m_\varphi$).

Lemma 1. *Let $\mathbb{G}$ act on $I = \{1 \dots m\}$, so that every $g \in \mathbb{G}$ corresponds to some permutation on $\{1 \dots m\}$. If $S^g = S$ and $J(x^g) = J(x)$ for every $g \in \mathbb{G}$ (i.e., $\mathbb{G}$ stabilizes both S and J) then the induced orbit partition* $\mathrm{Orb}_{\mathbb{G}}(I)$ *is a lifting partition for* $\inf_{x \in \mathcal{S}} J(x)$.

The second key observation is that all the above variational problems inherit the same symmetries of the parameter-tying exponential family, as captured in the lifting group $\mathbb{A}_\Delta[\mathcal{F}]$. Therefore, the lifting group will play the role of $\mathbb{G}$ in lemma 1 in lifting all of our variational problems.

Returning to (1), our general principle of lifted variational inference is captured in the following therem.

Theorem 4. *Let $\varphi = \varphi(\Delta) = \mathrm{Orb}_{\mathbb{A}_\Delta[\mathcal{F}]}(\mathcal{I})$. Then for all $\theta \in \Theta_\Delta$, φ is a lifting partition for (1), i.e.*

$$\sup_{\mu \in \mathcal{M}} \langle \theta, \mu \rangle - A^*(\mu) = \sup_{\mu \in \mathcal{M}_\varphi} \langle \theta, \mu \rangle - A^*(\mu) \tag{4}$$

Sktech of proof. From theorem 1, $\mathbb{A}[\mathcal{F}]$ stabilizes $\mathcal{M}$ and A^*; further, its subgroup $\mathbb{A}_\Delta(\mathcal{F})$ stabilizes every parameter $\theta \in \Theta_\Delta$. Thus, the lifting group $\mathbb{A}_\Delta(\mathcal{F})$ stabilizes both the constraint set and the objective function of (1). Invoking lemma 1, the induced orbit partition on $\mathcal{I}$ therefore yields a lifting partition.

In (4), we call the LHS the *ground* formulation of the variational problem, and the RHS the *lifted* formulation. Let $\ell = |\varphi|$ be the number of cells of φ, the *lifted* mean parameter space $\mathcal{M}_\varphi$ then effectively lies inside an ℓ-dimensional subspace where $\ell \leq m$. This forms the core of our principle of lifted variational inference: to perform optimization over the lower dimensional (and hopefully easier) constraint set $\mathcal{M}_\varphi$ instead of $\mathcal{M}$.

Remark. Because (1) has a unique solution $\mu = \mathbf{m}(\theta)$, theorem 4 implies that $\mathbf{m}(\Theta_\Delta) \subset \mathcal{M}_\varphi$. Further, the theorem also holds if we replace $\mathbb{A}_\Delta(\mathcal{F})$ with one of its subgroups $\mathbb{G}$: since $\varphi_\mathbb{G} = \mathrm{Orb}_\mathbb{G}(\mathcal{I})$ is finer than φ, it is obvious that $\varphi_\mathbb{G}$ is also a lifting partition. However, the smaller is the group $\mathbb{G}$, the finer is the lifting partition $\varphi_\mathbb{G}$, and the less symmetry can be exploited. In the extreme, $\mathbb{G}$ can be the trivial group, $\varphi_\mathbb{G}$ is the discrete partition putting each element of $\mathcal{I}$ in its own cell, and $\mathcal{M}_{\varphi_\mathbb{G}} = \mathcal{M}$, which corresponds to no lifting.

5.2 Characterization of $\mathcal{M}_\varphi$

We now give a characterization of the lifted mean parameter space $\mathcal{M}_\varphi$ in the case of discrete random variables. Note that $\mathcal{M}$ is the convex hull $\mathcal{M} = \mathrm{conv}\ \{\Phi(x) | x \in \mathcal{X}^n\}$ which is a polytope in $\mathbb{R}^m$, and $\mathbb{A}[\mathcal{F}]$ acts on the set of configurations $\mathcal{X}^n$ by the permuting action of π which maps $x \mapsto x^\pi$ for $x \in \mathcal{X}^n$.

Theorem 5. *Let $\mathcal{O} = \mathrm{Orb}_{\mathbb{A}_\Delta[\mathcal{F}]}(\mathcal{X}^n)$ be the set of $\mathcal{X}$-configuration orbits. For each orbit $\mathcal{C} \in \mathcal{O}$, let $\bar{\Phi}(\mathcal{C}) = \frac{1}{|\mathcal{C}|}\sum_{x\in\mathcal{C}} \Phi(x)$ be the feature-centroid of all the configurations in $\mathcal{C}$. Then $\mathcal{M}_{\varphi(\Delta)} = conv\ \{\bar{\Phi}(\mathcal{C}) | \mathcal{C} \in \mathcal{O}\}$.*

Thus, the *lifted polytope* $\mathcal{M}_\varphi$ can have at most $|\mathcal{O}|$ extreme points. The number of configuration orbits $|\mathcal{O}|$ can be much smaller than the total number of configurations $|\mathcal{X}|^n$ when the model is highly symmetric. For example, for a fully connected graphical model with identical pairwise and unary potentials and $\mathcal{X} = \{0, 1\}$ then every permutation $\pi \in \mathbb{S}_n$ is part of an automorphism; thus, every configuration with the same number of 1's belongs to the same orbit, and hence $|\mathcal{O}| = n + 1$. In general, however, $|\mathcal{O}|$ often is still exponential in n. We discuss approximations of $\mathcal{M}_\varphi$ in Section 6.

A representation of the lifted polytope $\mathcal{M}_\varphi$ by a set of constraints in $\mathbb{R}^{|\varphi|}$ can be directly obtained from the constraints of the polytope $\mathcal{M}$. First, we enforce the constraint $\mu \in \mathbb{R}^m_\varphi$: for each cell φ_j ($j = 1, \dots, |\varphi|$) of φ, let $\bar{\mu}_j$ be the common value of the variables μ_i, $i \in \varphi_j$. Let ρ be *the orbit mapping function* that maps each element $i \in \mathcal{I}$ to the corresponding cell $\rho(i) = j$ that contains i. Next, substituting μ_i by $\bar{\mu}_{\rho(i)}$ in the constraints of $\mathcal{M}$, we obtain a set of constraints in $\bar{\mu}$ (in vector form, we substitute μ by $D\bar{\mu}$ where $D_{ij} = 1$ if $i \in \varphi_j$ and 0 otherwise). In doing this, some constraints will become identical and thus redundant. In general, the number of non-redundant constraints can still be exponential.

5.3 Overcomplete Variational Problems

We now state analogous results in lifting the overcomplete variational problems (2) and (3) when $\mathcal{X}$ is finite. To simplify notation, we only present the case where features are unary or pairwise. As before, the lifting group $\mathbb{A}_\Delta[\mathcal{F}]$ will be used to induce a lifting partition. However, we need to define the action of this group on the set of overcomplete features $\mathcal{I}^o$.

For each automorphism $(\pi, \gamma) \in \mathbb{A}[\mathcal{F}]$, γ gives us the permutation on $\mathcal{I}$. In order to obtain a permutation on $\mathcal{I}^o$, we will need to use π. By theorem 1, π is an automorphism of the graphical model graph $\mathcal{G}$. Since overcomplete features naturally correspond to nodes and edges of $\mathcal{G}$, π induces a natural bijection on $\mathcal{I}^o$ that maps $v{:}t \mapsto \pi(v){:}t$ and $\{u{:}t, v{:}t'\} \mapsto \{\pi(u){:}t, \pi(v){:}t'\}$. Define $\varphi^o = \varphi^o(\Delta) = \mathrm{Orb}_{\mathbb{A}_\Delta[\mathcal{F}]}(\mathcal{I}^o)$ to be the orbits of $\mathbb{A}_\Delta[\mathcal{F}]$ acting on the set of overcomplete features. Then

Theorem 6. *For all $\theta \in \Theta_\Delta$, φ^o is a lifting partition for the variational problems (2) and (3).*

Thus, the optimization domain can be restricted to $\mathcal{M}^o_{\varphi^o}$ which we term the *lifted marginal polytope*. The cells of φ^o are intimately connected to the node, edge and arc orbits of the graph $\mathcal{G}$ induced by $\mathbb{A}_\Delta[\mathcal{F}]$. We now list all the cells of φ^o in the case where $\mathcal{X} = \{0, 1\}$: each node orbit $\mathbf{v}$ corresponds to 2 cells $\{v : t | v \in \mathbf{v}\}$, $t \in \{0, 1\}$; each edge orbit $\mathbf{e}$ corresponds to 2 cells $\{\{u : t, v : t\} \mid \{u, v\} \in \mathbf{e}\}$, $t \in \{0, 1\}$; and each arc orbit $\mathbf{a}$ corresponds to the cell $\{\{u : 0, v : 1\} | (u, v) \in \mathbf{a}\}$. The orbit mapping function ρ maps each element of $\mathcal{I}^o$ to its orbit as follows: $\rho(v{:}t) = \overline{v}{:}t$, $\rho(\{u{:}t, v{:}t\}) = \{\overline{u, v}\}{:}t$, $\rho(\{u{:}0, v{:}1\}) = (\overline{u, v}){:}01$ where $\overline{v}$ represents the node-orbit of v, $\{\overline{u, v}\}$ represents the edge-orbit of $\{u, v\}$ and $(\overline{u, v})$ represents the arc-orbit of (u, v).

The total number of cells of φ^o is $2|\bar{V}| + 2|\bar{E}| + |\bar{A}|$ where $|\bar{V}|$, $|\bar{E}|$ and $|\bar{A}|$ are the number of node, edge and arc orbits of $\mathcal{G}$ (note that $|\bar{A}| \le 2|\bar{E}|$). Therefore, in working with $\mathcal{M}^o_{\varphi^o}$, the big-$O$ order of the number of variables is reduced from the number of nodes and edges in $\mathcal{G}$ to the number of node and edge orbits.

For MAP inference, (3) is equivalent to the lifted problem $\sup_{\mu^o \in \mathcal{M}^o_{\varphi^o}} \langle \theta^o, \mu^o \rangle$. A single ground MAP solution $\hat{x}$ leads to an entire configuration orbit $\mathcal{C} = \mathrm{orb}_{\mathbb{A}_\Delta[\mathcal{F}]}(\hat{x})$ of MAP solutions. The feature-centroid $\bar{\mu}^o = \bar{\Phi}^o(\mathcal{C}) = \frac{1}{|\mathcal{C}|}\sum_{x\in\mathcal{C}} \Phi^o(x)$ then lies inside $\mathcal{M}^o_{\varphi^o}$ and is the corresponding lifted MAP solution. Furthermore, $\bar{\mu}^o_{\mathbf{v}:t} = \frac{1}{|\mathbf{v}|}\sum_{v' \in \mathbf{v}} \phi^o_{v':t}(\hat{x})$ is the fraction of the ground variables in $\hat{x}_\mathbf{v}$ assigned the value t, and similarly for pairwise features. Note that from the learning (parameter estimation) point of view, the lifted MAP solution is more useful than any single MAP solution alone.

6 Lifted Approximate MAP Inference

Approximate convex variational inference typically works with a tractable convex approximation of $\mathcal{M}$ and a tractable convex approximation of the negative entropy function A^*. In this paper we consider only lifted outer bounds of $\mathcal{M}^o$ (and thus restrict ourselves to the discrete case). We leave the problem of handling approximations of A^* to future work. Our focus is the LP relaxation of the MAP inference problem (3) and its lifted formulation.

To find an approximate lifted solution, since any outer bound OUTER $\supset \mathcal{M}^o$ yields an outer bound OUTER$_{\varphi^o}$ of $\mathcal{M}^o_{\varphi^o}$, we can always relax the lifted problem and replace $\mathcal{M}_{\varphi^o}$ by OUTER$_{\varphi^o}$. But is the relaxed lifted problem on OUTER$_{\varphi^o}$ equivalent to the relaxed ground problem on OUTER? This depends on whether φ^o is a lifting partition for the relaxed ground problem.

Theorem 7. *If the set OUTER = OUTER($\mathcal{G}$) depends only on the graphical model structure $\mathcal{G}$ of $\mathcal{F}$, then $\forall\theta \in \Theta_\Delta$, φ^o is a lifting partition for the relaxed MAP problem*

$$\sup_{\mu^o \in OUTER} \langle \theta^o, \mu^o \rangle = \sup_{\mu^o \in OUTER_{\varphi^o}} \langle \theta^o, \mu^o \rangle$$

The most often used outer bound of $\mathcal{M}^o$ is the local marginal polytope LOCAL($\mathcal{G}$) [19], which enforces consistency for marginals on nodes and between nodes and edges of $\mathcal{G}$. [17, 18] used CYCLE($\mathcal{G}$), which is a tighter bound that also enforces consistency of edge marginals on the same cycle of $\mathcal{G}$. The Sherali-Adams hierarchy[3] [15] provides a sequence of outer bounds of $\mathcal{M}^o$, starting from LOCAL($\mathcal{G}$) and progressively tightening it to the exact marginal polytope $\mathcal{M}^o$. All of these outer bounds depend only on the structure of the graphical model $\mathcal{G}$, and thus the corresponding relaxed MAP problems admit φ^o as a lifting partition. Note that with the exception when OUTER = LOCAL, equitable partitions [6] of $\mathcal{G}$ such as those used in [11] are not lifting partitions for the approximate variational problem in theorem 7.[4]

7 Lifted MAP Inference on the Local Polytope

We now focus on lifted approximate MAP inference using the local marginal polytope LOCAL. From this point on, we also restrict ourselves to models where the features are pairwise or unary, and the variables are binary ($\mathcal{X} = \{0, 1\}$).

We first aim to give an explicit characterization of the constraints of the lifted local polytope LOCAL$_{\varphi^o}$. The local polytope LOCAL($\mathcal{G}$) is defined as the set of locally consistent pseudo-marginals.

$$\left\{ \tau \geq 0 \;\middle|\; \begin{array}{ll} \tau_{v:0} + \tau_{v:1} = 1 & \forall v \in \mathcal{V}(\mathcal{G}) \\ \tau_{\{u:0,v:0\}} + \tau_{\{u:0,v:1\}} = \tau_{u:0} & \\ \tau_{\{u:0,v:0\}} + \tau_{\{v:0,u:1\}} = \tau_{v:0} & \forall \{u,v\} \in E(\mathcal{G}) \\ \tau_{\{u:1,v:1\}} + \tau_{\{u:0,v:1\}} = \tau_{v:1} & \\ \tau_{\{u:1,v:1\}} + \tau_{\{v:0,u:1\}} = \tau_{u:1} & \end{array} \right\}$$

Substituting τ_i by the corresponding $\bar{\tau}_{\rho(i)}$ where $\rho()$ is given in subsection 5.3, and by noting that constraints generated by $\{u, v\}$ in the same edge orbits are redundant, we obtain the constraints for the *lifted local polytope* LOCAL$_{\varphi^o}$ as follows.

$$\left\{ \bar{\tau} \geq 0 \;\middle|\; \begin{array}{ll} \bar{\tau}_{\mathbf{v}:0} + \bar{\tau}_{\mathbf{v}:1} = 1 & \forall \text{ node orbit } \mathbf{v} \\ \bar{\tau}_{\mathbf{e}:00} + \bar{\tau}_{(\overline{u,v}):01} = \bar{\tau}_{\bar{u}:0} & \\ \bar{\tau}_{\mathbf{e}:00} + \bar{\tau}_{(\overline{v,u}):01} = \bar{\tau}_{\bar{v}:0} & \forall \text{ edge orbit } \mathbf{e} \text{ with} \\ \bar{\tau}_{\mathbf{e}:11} + \bar{\tau}_{(\overline{u,v}):01} = \bar{\tau}_{\bar{v}:1} & \{u,v\} \text{ a representative of } \mathbf{e} \\ \bar{\tau}_{\mathbf{e}:11} + \bar{\tau}_{(\overline{v,u}):01} = \bar{\tau}_{\bar{u}:1} & \end{array} \right\}$$

[3]A note about terminology: Following the tradition in lifted inference, this paper uses the term *lift* to refer to the exploitation of symmetry for avoiding doing inference on the *ground* model. It is unfortunate that the term *lift* has also been used in the context of coming up with better bounds for the marginal polytopes. There, *lift* (as in lift-and-project) means to move to a higher dimensional space where constraints can be more easily expressed with auxiliary variables.

[4]As a counter example, consider a graphical model whose structure is the Frucht graph (Fig. 1(d)). Since this is a regular graph, LOCAL approximation yields identical constraints for every node. However, the nodes on this graph participate in cycles of different length, hence are subject to different cycle constraints.

Thus, the number of constraints needed to describe the lifted local polytope LOCAL$_{\varphi^o}$ is $O(|\bar{V}| + |\bar{E}|)$. Similar to the ground problem, these constraints can be derived from a graph representation of the node and edge orbits. Define the *lifted graph* $\bar{\mathcal{G}}$ to be a graph whose nodes are the set of node orbits $\bar{V}$ of $\mathcal{G}$. For each edge orbit $\mathbf{e}$ with a representative $\{u, v\} \in \mathbf{e}$, there is a corresponding edge on $\bar{\mathcal{G}}$ that connects the two node orbits $\bar{u}$ and $\bar{v}$. Note that unlike $\mathcal{G}$, the lifted graph $\bar{\mathcal{G}}$ in general is not a simple graph and can contain self-loops and multi-edges between two nodes. Figure 2(a) and (c) show the ground graphical model $\mathcal{G}$ and the lifted graph $\bar{\mathcal{G}}$ for the example 1.

Next consider the linear objective function $\langle \theta^o, \tau \rangle$. Substituting τ_i by the corresponding $\bar{\tau}_{\rho(i)}$, we can rewrite the objective function in terms of $\bar{\tau}$ as $\langle \bar{\theta}, \bar{\tau} \rangle$ where the coefficients $\bar{\theta}$ are defined on nodes and edges of the lifted graph $\bar{\mathcal{G}}$ as follows. For each node orbit $\mathbf{v}$, $\bar{\theta}_{\mathbf{v}:t} = \sum_{v' \in \mathbf{v}} \theta^o_{v':t} = |\bar{v}|\theta^o_{v:t}$ where $t \in \{0, 1\}$ and v is any representative member of $\mathbf{v}$. For each edge orbit $\mathbf{e}$ with a representative $\{u, v\} \in \mathbf{e}$, $\bar{\theta}_{\mathbf{e}:tt} = \sum_{\{u',v'\} \in \mathbf{e}} \theta^o_{\{u':t,v':t\}} = |\mathbf{e}|\theta^o_{\{u:t,v:t\}}$ where $t \in \{0, 1\}$, $\bar{\theta}_{(\overline{u,v}):01} = \sum_{(u',v') \in (\overline{u,v})} \theta^o_{\{u':0,v':1\}} = |(\overline{u,v})|\theta^o_{\{u:0,v:1\}}$. Note that typically the two arc-orbits $(\overline{u,v})$ and $(\overline{v,u})$ are not the same, in which case $|(\overline{u,v})| = |(\overline{v,u})| = |\mathbf{e}|$. However, in the case $(\overline{u,v}) = (\overline{v,u})$, then $|(\overline{u,v})| = |(\overline{v,u})| = 2|\mathbf{e}|$.

We have shown that the lifted formulation for MAP inference on the local polytope can be described in terms of the lifted variables $\bar{\tau}$ and the lifted parameters $\bar{\theta}$. These lifted variables and parameters are associated with the orbits of the ground graphical model. Thus, the derived lifted formulation can also be read out directly from the lifted graph $\bar{\mathcal{G}}$. In fact, the derived lifted formulation is the local relaxed MAP problem of the lifted graphical model $\bar{\mathcal{G}}$. Therefore, any algorithm for solving the local relaxed MAP problem on $\mathcal{G}$ can also be used to solve the derived lifted formulation on $\bar{\mathcal{G}}$. For example, performing coordinate descent in the dual formulation [5] of the lifted local LP yields the lifted MPLP. Note that MPLP is an asynchronous message passing algorithms that cannot be lifted by grouping identical messages.

8 Beyond Local Polytope: Lifted MAP Inference with Cycle Inequalities

We now discuss lifting the MAP relaxation on CYCLE($\mathcal{G}$), a bound obtained by tightening LOCAL($\mathcal{G}$) with an additional set of linear constraints that hold on cycles of the graphical model structure $\mathcal{G}$, called cycle constraints [17]. These constraints mean the number of cuts (transitions from 0 to 1 or vice versa) in any configuration on a cycle of $\mathcal{G}$ must be even. Cycle constraints can be expressed as linear constraints as follows. For every cycle C (set of edges that form a cycle in $\mathcal{G}$) and every odd-sized subset $F \subseteq C$

$$\sum_{\{u,v\} \in F} nocut(\{u,v\}, \tau) + \sum_{\{u,v\} \in C \setminus F} cut(\{u,v\}, \tau) \geq 1 \tag{5}$$

where $nocut(\{u,v\},\tau) = \tau_{\{u:0,v:0\}} + \tau_{\{u:1,v:1\}}$ and $cut(\{u,v\},\tau) = \tau_{\{u:0,v:1\}} + \tau_{\{v:0,u:1\}}$.

Theorem 7 guarantees that MAP inference on CYCLE can be lifted by restricting the feasible domain to CYCLE$_{\varphi^o}$, which we term the *lifted cycle polytope*. Substituting the original variables τ by the lifted variables $\bar{\tau}$, we obtain the *lifted cycle constraints* in terms of $\bar{\tau}$

$$\sum_{\{u,v\}\in F} nocut(\{\overline{u,v}\},\bar{\tau}) + \sum_{\{u,v\}\in C\setminus F} cut(\{\overline{u,v}\},\bar{\tau}) \geq 1 \tag{6}$$

where $nocut(\{\overline{u,v}\},\bar{\tau}) = \bar{\tau}_{\{\overline{u,v}\}:00} + \bar{\tau}_{\{\overline{u,v}\}:11}$ and $cut(\{\overline{u,v}\},\bar{\tau}) = \bar{\tau}_{(\overline{u,v}):01} + \bar{\tau}_{(\overline{v,u}):01}$ where $(\overline{u,v})$ and $(\overline{v,u})$ are the arc-orbits corresponding to the node-orbit $\{\overline{u,v}\}$.

8.1 Lifted Cycle Constraints on All Cycles Passing Through a Fixed Node

It is not possible to extract all lifted cycle constraints just by examining the lifted graphical model $\bar{\mathcal{G}}$ since there could be cycles in $\bar{\mathcal{G}}$ that do not correspond to any cycles in $\mathcal{G}$. However, we can characterize all constraints on all cycles passing through a fix node i in $\mathcal{G}$.

Let Cyc$[i]$ be the set of (ground) cycle constraints generated from all cycles passing through i. A cycle is simple if it does not intersect with itself or contain repeated edges; [17] considers only simple cycles, but we will also consider any cycle, including non-simple cycles in Cyc$[i]$. Adding non-simple cycles to the mix does not change the story since constraints on non-simple cycles of $\mathcal{G}$ are redundant. We now give a precise characterization of $\overline{\text{Cyc}}[i]$, the set of lifted cycle constraints obtained by lifting all cycle constraints in Cyc$[i]$ via the transformation from (5) to (6).

The lifted graph fixing i, $\bar{\mathcal{G}}[i]$ is defined as follows. Let $\mathbb{A}_\Delta[\mathcal{F},i]$ be the subgroup of $\mathbb{A}_\Delta[\mathcal{F}]$ that fixes i, that is $\pi(i) = i$. The set of nodes of $\bar{\mathcal{G}}[i]$ is the set of node orbits $\bar{V}[i]$ of $\mathcal{G}$ induced by $\mathbb{A}_\Delta[\mathcal{F},i]$, and the set of edges is the set of edge orbits $\bar{E}[i]$ of $\mathcal{G}$. Each edge orbit connects to the orbits of the two adjacent nodes (which could form just one node orbit). Since i is fixed, $\{i\}$ is a node orbit, and hence is a node on $\bar{\mathcal{G}}[i]$. Note that $\bar{\mathcal{G}}[i]$ in general is not a simple graph: it can have multi-edges and loops.

Theorem 8. *Let $\bar{C}$ be a cycle (not necessarily simple) in $\bar{\mathcal{G}}[i]$ that passes through the node $\{i\}$. For any odd-sized $\bar{F} \subset \bar{C}$*

$$\sum_{\mathbf{e}\in\bar{F}} nocut(\mathbf{e},\bar{\tau}) + \sum_{\mathbf{e}\in\bar{C}\setminus\bar{F}} cut(\mathbf{e},\bar{\tau}) \geq 1 \tag{7}$$

is a constraint in $\overline{\text{Cyc}}[i]$. Further, all constraints in $\overline{\text{Cyc}}[i]$ can be expressed this way.

8.2 Separation of Lifted Cycle Constraints

While the number of cycle constraints may be reduced significantly in the lifted space, it may still be computationally expensive to list all of them. To address this issue, we follow [17] and employ a cutting plane approach in which we find and add only the most violated lifted cycle constraint in each iteration (separation operation).

For finding the most violated lifted cycle constraint, we propose a lifted version of the method presented by [17], which performs the separation by iterating over the nodes of the graph $\mathcal{G}$ and for each node i finds the most violated cycle constraint from all cycles passing through i. Theorem 8 suggests that all lifted cycle constraints in $\overline{\text{Cyc}}[i]$ can be separated by mirroring $\bar{\mathcal{G}}[i]$ and performing a shortest path search from $\{i\}$ to its mirrored node, similar to the way separation is performed on ground cycle constraints [17].

To find the most violated lifted cycle constraint, we could first find the most violated lifted cycle constraint C_i in $\overline{\text{Cyc}}[i]$ for each node i, and then take the most violated constraints over all C_i. However, note that if i and i' are in the same node orbit, then $\overline{\text{Cyc}}[i] = \overline{\text{Cyc}}[i']$. Hence, we can perform separation using the following algorithm:

1. For each node orbit $\bar{v} \in \bar{V}$, choose a representative $i \in \bar{v}$ and find its most violated lifted cycle constraint $C_{\bar{v}} \in \overline{\text{Cyc}}[i]$ using a shortest path algorithm on the mirror graph of $\bar{\mathcal{G}}[i]$.
2. Return the most violated constraint over all $C_{\bar{v}}$.

Notice that both $\bar{\mathcal{G}}[i]$ and its mirror graph have to be calculated only once per graph. In each separation iteration we can reuse these structures, provided that we adapt the edge weights in the mirror graph according to the current marginals.

9 Experiments

First, we evaluate methods for detecting symmetries described in Section 4 on the "Friends & Smokers" MLN[5] [16]. The first method (*nauty)* grounds the MLN then finds a lifting partition. The second (*renaming*) does not require grounding, but uses the renaming group to find a lifting partition. Table 1 presents the results for varying domain sizes where for a random 10% of all people it is known whether they smoke or not. Although *nauty* finds a more compact lifted graph, it takes significantly more time than using the renaming group. For this reason, our subsequent experiment only makes use of the renaming group and orbits.[6]

Figure 4 shows the run time performance of MAP inference using local and cycle LP formulations (both ground and lifted algorithms use the off-the-shelf Gurobi LP solver). For cutting plane, we use the in-out variant [1] with parameter $\alpha = 0.99$ to improve convergence. All lifted variants are several order-of-magnitude faster than their ground counterparts. We also find that for this particular MLN, all solutions found by the local LP formulation immediately satisfy all the cycle constraints. Closer examination reveals that this MLN prescribes attractive potentials on the pairs $(Smoke(x), Smoke(y))$, thus MAP

[5]The ground graphical model of this MLN has tree-width equals to the domain size.

[6]Independent result reported in [13] seems to suggest better performance can be obtained using SAUCY, a more modern tool for finding graph automorphism.

Table 1: Symmetries detection on the Friends & Smokers MLN with 10% known people. * means the process did not finish within a day.

		10	20	50	100	200	1000
Nauty	#Orbits	12	23	25	27	*	*
	Time(s)	.49	1.77	172.79	9680.48	*	*
Renaming	#Orbits	12	23	80	255	905	20505
	Time(s)	.08	.09	.221	.4	.84	2.19

assignments to unknown smokers are either all true or all false.

Next, we conduct experiments with the following "Lovers & Smokers" MLN.

100	$Male(x) \Leftrightarrow \neg Female(x)$
2	$Male(x) \wedge Smokes(x)$
2	$Female(x) \wedge \neg Smokes(x)$
0.5	$x \neq y \wedge Male(x) \wedge Female(y) \wedge Loves(x,y)$
0.5	$x \neq y \wedge Loves(x,y) \Rightarrow (Smokes(x) \Leftrightarrow Smokes(y))$
-100	$x \neq y \wedge y \neq z \wedge z \neq x \wedge Loves(x,y) \wedge Loves(y,z) \wedge Loves(x,z)$

Note that this model is much more difficult because the last formula has a repulsive potential and is fully transitive. As far as we know, to date, no exact lifted inference algorithms can handle transitive clauses in polynomial time.

The first experiment assumes no evidence, a situation commonly encountered during the inference step [9] of any perceptron-style generative parameter learning method. As before, we compare local and cycle LP formulations, both ground and lifted while varying the domain size of the MLN. Figure 5(a) shows the lifted variants achieve *constant* running time regardless of the actual domain size, and are significantly more efficient than their ground counterparts as the domain size increase. Figure 5(b) illustrates how the objective value changes over cutting plane iterations (and hence time), for domain size $= 5$. Both the local polytope (ground and lifted) approaches have no cutting plane iterations, and hence are represented as single points. We use Integer Linear Programming (ILP) to compute a reference point of the lowest possible optimal objective value. Notice all methods are based on outer/upper bounds on the variational objective, and hence are decreasing over time. First, we can observe that the CYCLE methods converge to a solution substantially better than the LOCAL methods. However, although lifted CYCLE converges quickly, the ground CYCLE algorithm converges very slowly.

The second experiment varies the number of observed constants with random soft evidence while fixing the domain size to 100. Because ground methods do not scale to this size, we only compare lifted LOCAL and lifted CYCLE. Figure 6 shows both the running time and the obtained objective value. Observe that lifted CYCLE significantly improves the MAP objective value but at a significant computational cost when the number of observed constants increases. We note that with soft evidence, the lifted model essentially becomes a ground model which contains a large number of cycles induced by the transitive clause in the model.

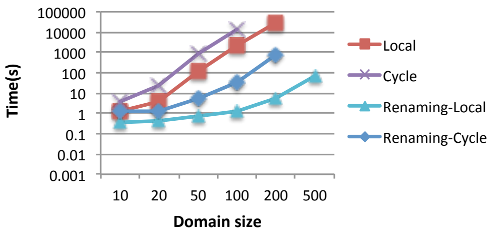


Figure 4: (Best viewed in color) "Friends & Smokers" MLN with 10% known people.

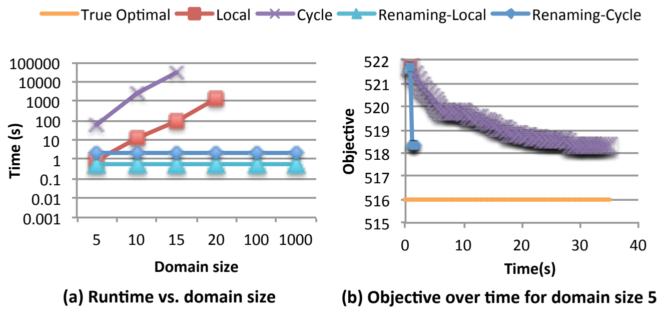


Figure 5: (Best viewed in color) "Lovers & Smokers" MLN without evidence. The local and cycle methods did not finish within a day for larger domain sizes.

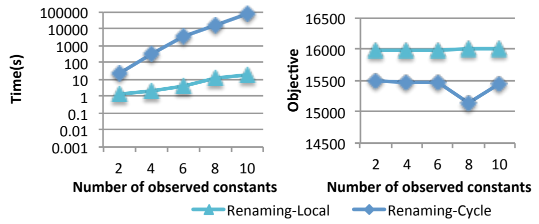


Figure 6: "Lovers & Smokers" MLN with random soft evidence, domain size = 100.

10 Conclusion

We presented a new general framework for lifted variational inference by introducing and studying a precise mathematical definition of symmetry of graphical models via the construction of their automorphism groups. Using the device of automorphism groups, orbits of random variables are obtained, and lifted variational inference materializes as performing the corresponding convex variational optimization problem in the space of per-orbit random variables. Our framework enables lifting a large class of approximate variational MAP inference algorithms, including the first lifted algorithm for MAP inference with cycle constraints. We presented experimental results demonstrating the clear benefits of the lifted over the ground formulations. Future extension includes how to handle approximations of the convex upper-bounds of negative entropy function A^*, which would enable lifting the full class of approximate convex variational marginal inference.

Acknowledgement. The authors gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA, AFRL, or the U.S. government.

References

[1] Walid Ben-Ameur and Jose Neto. Acceleration of cutting-plane and column generation algoirthms: applications to network design. *Networks*, 49(1):3 – 17, 2007.

[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. Technical report, 2012. URL `http://arxiv.org/abs/1207.4814`.

[4] 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.

[5] Amir Globerson and Tommi Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Advances in Neural Information Processing Systems (NIPS '07)*, pages 553–560, 2007.

[6] Chris Godsil and Gordon Royle. *Algebraic Graph Theory*. Springer, 2001.

[7] V. Gogate and P. Domingos. Exploiting logical structure in lifted probabilistic inference. In *AAAI Workshop on Statistical Relational AI*, 2010.

[8] 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.

[9] 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.

[10] Brendan D. McKay. Practical Graph Isomorphism. *Congressus Numerantium*, 30:45–87, 1981.

[11] M. Mladenov, B. Ahmadi, and K. Kersting. Lifted linear programming. In *15th International Conference on Artificial Intelligence and Statistics (AISTATS 2012)*, 2012.

[12] Mathias Niepert. Lifted probabilistic inference: an MCMC perspective. In *Statistical Relational AI Workshop at UAI 2012*, 2012.

[13] Mathias Niepert. Markov chains on orbits of permutation groups. In *UAI-2012*, 2012.

[14] Matt Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

[15] Hanif D. Sherali and Warren 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.

[16] 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.

[17] D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In *Advances in Neural Information Processing Systems (NIPS '07)*, pages 1393–1400, 2007.

[18] David Sontag. *Approximate Inference in Graphical Models using LP Relaxations*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2010.

[19] Martin Wainwright and Michael Jordan. *Graphical Models, Exponential Families, and Variational Inference.* Now Publishers, 2008.

POMDPs under Probabilistic Semantics

Krishnendu Chatterjee
IST Austria
krishnendu.chatterjee@ist.ac.at

Martin Chmelík
IST Austria
martin.chmelik@ist.ac.at

Abstract

We consider partially observable Markov decision processes (POMDPs) with limit-average payoff, where a reward value in the interval $[0, 1]$ is associated to every transition, and the payoff of an infinite path is the long-run average of the rewards. We consider two types of path constraints: (i) quantitative constraint defines the set of paths where the payoff is at least a given threshold $\lambda_1 \in (0, 1]$; and (ii) qualitative constraint which is a special case of quantitative constraint with $\lambda_1 = 1$. We consider the computation of the almost-sure winning set, where the controller needs to ensure that the path constraint is satisfied with probability 1. Our main results for qualitative path constraint are as follows: (i) the problem of deciding the existence of a finite-memory controller is EXPTIME-complete; and (ii) the problem of deciding the existence of an infinite-memory controller is undecidable. For quantitative path constraint we show that the problem of deciding the existence of a finite-memory controller is undecidable.

1 Introduction

Partially observable Markov decision processes (POMDPs). *Markov decision processes (MDPs)* are standard models for probabilistic systems that exhibit both probabilistic and nondeterministic behavior [10]. MDPs have been used to model and solve control problems for stochastic systems [7, 23]: nondeterminism represents the freedom of the controller to choose a control action, while the probabilistic component of the behavior describes the system response to control actions. In *perfect-observation (or perfect-information) MDPs (PIMDPs)* the controller can observe the current state of the system to choose the next control actions, whereas in *partially observable MDPs (POMDPs)* the state space is partitioned according to observations that the controller can observe, i.e., given the current state, the controller can only view the observation of the state (the partition the state belongs to), but not the precise state [20]. POMDPs provide the appropriate model to study a wide variety of applications such as in computational biology [5], speech processing [19], image processing [4], robot planning [13, 11], reinforcement learning [12], to name a few. POMDPs also subsume many other powerful computational models such as probabilistic finite automata (PFA) [24, 21] (since probabilistic finite automata (aka blind POMDPs) are a special case of POMDPs with a single observation).

Limit-average payoff. A *payoff* function maps every infinite path (infinite sequence of state action pairs) of a POMDP to a real value. The most well-studied payoff in the setting of POMDPs is the *limit-average* payoff where every state action pair is assigned a real-valued reward in the interval $[0, 1]$ and the payoff of an infinite path is the long-run average of the rewards on the path [7, 23]. POMDPs with limit-average payoff provide the theoretical framework to study many important problems of practical relevance, including probabilistic planning and several stochastic optimization problems [11, 2, 16, 17, 27].

Expectation vs probabilistic semantics. Traditionally, MDPs with limit-average payoff have been studied with the *expectation* semantics, where the goal of the controller is to maximize the expected limit-average payoff. The expected payoff value can be $\frac{1}{2}$ when with probability $\frac{1}{2}$ the payoff is 1, and with remaining probability the payoff is 0. In many applications of system analysis (such as robot planning and control) the relevant question is the probability measure of the paths that satisfy certain criteria, e.g., whether the probability measure of the paths such that the limit-average payoff is 1 (or the payoff is at least

$\frac{1}{2}$) is at least a given threshold (e.g., see [1, 13]). We classify the path constraints for limit-average payoff as follows: (1) *quantitative constraint* that defines the set of paths with limit-average payoff at least λ_1, for a threshold $\lambda_1 \in (0, 1]$; and (2) *qualitative constraint* is the special case of quantitative constraint that defines the set of paths with limit-average payoff 1 (i.e., the special case with $\lambda_1 = 1$). We refer to the problem where the controller must satisfy a path constraint with a probability threshold $\lambda_2 \in (0, 1]$ as the *probabilistic* semantics. An important special case of probabilistic semantics is the *almost-sure* semantics, where the probability threshold is 1. The almost-sure semantics is of great importance because there are many applications where the requirement is to know whether the correct behavior arises with probability 1. For instance, when analyzing a randomized embedded scheduler, the relevant question is whether every thread progresses with probability 1. Even in settings where it suffices to satisfy certain specifications with probability $\lambda_2 < 1$, the correct choice of λ_2 is a challenging problem, due to the simplifications introduced during modeling. For example, in the analysis of randomized distributed algorithms it is quite common to require correctness with probability 1 (e.g., [22, 26]). Besides its importance in practical applications, almost-sure convergence, is a fundamental concept in probability theory, and provide stronger convergence guarantee than convergence in expectation [6].

Previous results. There are several deep undecidability results established for the special case of probabilistic finite automata (PFA) (that immediately imply undecidability for POMDPs). The basic undecidability results are for PFA over finite words: The emptiness problem for PFA under probabilistic semantics is undecidable over finite words [24, 21, 3]; and it was shown in [16] that even the following approximation version is undecidable: for any fixed $0 < \epsilon < \frac{1}{2}$, given a probabilistic automaton and the guarantee that either (a) there is a word accepted with probability at least $1-\epsilon$; or (ii) all words are accepted with probability at most ϵ; decide whether it is case (i) or case (ii). The almost-sure problem for probabilistic automata over finite words reduces to the non-emptiness question of universal automata over finite words and is PSPACE-complete. However, another related decision question whether for every $\epsilon > 0$ there is a word that is accepted with probability at least $1 - \epsilon$ (called the value 1 problem) is undecidable for probabilistic automata over finite words [8]. Also observe that all undecidability results for probabilistic automata over finite words carry over to POMDPs where the controller is restricted to finite-memory strategies. The importance of finite-memory strategies in applications has been established in [9, 14, 18].

Table 1: Complexity: New results are in bold fonts

	Almost-sure semantics		Prob. semantics
	Fin. mem.	Inf. mem.	Fin./Inf. mem.
PFA	PSPACE-c	PSPACE-c	Undec.
POMDP Qual. Constr.	**EXPTIME-c**	**Undec.**	Undec.
POMDP Quan. Constr.	**Undec.**	**Undec.**	Undec.

Our contributions. Since under the general probabilistic semantics, the decision problems are undecidable even for PFA, we consider POMDPs with limit-average payoff under the almost-sure semantics. We present a complete picture of decidability as well as optimal complexity.

(Almost-sure winning for qualitative constraint). We first consider limit-average payoff with qualitative constraint under almost-sure semantics. We show that *belief-based* strategies are not sufficient (where a belief-based strategy is based on the subset construction that remembers the possible set of current states): we show that there exist POMDPs with limit-average payoff with qualitative constraint where finite-memory almost-sure winning strategy exists but there exists no belief-based almost-sure winning strategy. Our counter-example shows that standard techniques based on subset construction (to construct an exponential size PIMDP) are not adequate to solve the problem. We then show one of our main result that given a POMDP with $|S|$ states and $|\mathcal{A}|$ actions, if there is a finite-memory almost-sure winning strategy to satisfy the limit-average payoff with qualitative constraint, then there is an almost-sure winning strategy that uses at most $2^{3\cdot|S|+|\mathcal{A}|}$ memory. Our exponential memory upper bound is asymptotically optimal, as even for PFA over finite words, exponential memory is required for almost-sure winning (follows from the fact that the shortest witness word for non-emptiness of universal finite automata is at least exponential). We then show that the problem of deciding the existence of a finite-memory almost-sure winning strategy for limit-average payoff with qualitative constraint is EXPTIME-complete for POMDPs. In contrast to our result for finite-memory strategies, we show that deciding the existence of an infinite-memory almost-sure winning strategy for limit-average payoff with qualitative constraint is undecidable for POMDPs.

(Almost-sure winning with quantitative constraint). In contrast to our decidability result under finite-memory strategies for qualitative constraint, we show that the almost-sure winning problem for limit-average payoff with quantitative constraint is undecidable even for finite-memory strategies for POMDPs.

In summary we establish the precise decidability frontier for POMDPs with limit-average payoff under probabilistic semantics (see Table 1). For practical purposes, the most prominent question is the problem of finite-memory strategies, and for finite-memory strategies we establish decidability with EXPTIME-complete complexity for the important special case of qualitative constraint under almost-sure semantics.

2 Definitions

We present the definitions of POMDPs, strategies, objectives, and other basic notions required for our results. We follow standard notations from [23, 15].

Notations. Given a finite set X, we denote by $\mathcal{P}(X)$ the set of subsets of X, i.e., $\mathcal{P}(X)$ is the power set of X. A probability distribution f on X is a function $f : X \to [0, 1]$ such that $\sum_{x\in X} f(x) = 1$, and we denote by $\mathcal{D}(X)$ the set of all probability distributions on X. For $f \in \mathcal{D}(X)$ we denote by $\mathrm{Supp}(f) = \{x \in X \mid f(x) > 0\}$ the support of f.

Definition 1 (POMDP). *A* Partially Observable Markov Decision Process (POMDP) *is a tuple $G = (S, \mathcal{A}, \delta, \mathcal{O}, \gamma, s_0)$ where: (i) S is a finite set of states; (ii) $\mathcal{A}$ is a finite alphabet of* actions*; (iii) $\delta : S \times \mathcal{A} \to \mathcal{D}(S)$ is a* probabilistic transition function *that given a state s and an action $a \in \mathcal{A}$ gives the probability distribution over the successor states, i.e., $\delta(s, a)(s')$ denotes the transition probability from s to s' given action a; (iv) $\mathcal{O}$ is a finite set of* observations*; (v) $\gamma : S \to \mathcal{O}$ is an* observation function *that maps every state to an observation; and (vi) s_0 is the initial state.*

Given $s, s' \in S$ and $a \in \mathcal{A}$, we also write $\delta(s'|s, a)$ for $\delta(s, a)(s')$. A state s is *absorbing* if for all actions a we have $\delta(s, a)(s) = 1$ (i.e., s is never left from s). For an observation o, we denote by $\gamma^{-1}(o) = \{s \in S \mid \gamma(s) = o\}$ the set of states with observation o. For a set $U \subseteq S$ of states and $O \subseteq \mathcal{O}$ of observations we denote $\gamma(U) = \{o \in \mathcal{O} \mid \exists s \in U.\ \gamma(s) = o\}$ and $\gamma^{-1}(O) = \bigcup_{o\in O} \gamma^{-1}(o)$.

Plays and belief-updates. A *play* (or a path) in a POMDP is an infinite sequence $(s_0, a_0, s_1, a_1, s_2, a_2, \ldots)$ of states and actions such that for all $i \geq 0$ we have $\delta(s_i, a_i)(s_{i+1}) > 0$. We write Ω for the set of all plays. For a finite prefix $w = (s_0, a_0, s_1, a_1, \ldots, s_n)$ we denote by $\gamma(w) = (\gamma(s_0), a_0, \gamma(s_1), a_1, \ldots, \gamma(s_n))$ the observation and action sequence associated with w. For a finite sequence $\rho = (o_0, a_0, o_1, a_1, \ldots, o_n)$ of observations and actions, the *belief* $\mathcal{B}(\rho)$ after the prefix ρ is the set of states in which a finite prefix of a play can be after the sequence ρ of observations and actions, i.e., $\mathcal{B}(\rho) = \{s_n = \mathsf{Last}(w) \mid w = (s_0, a_0, s_1, a_1, \ldots, s_n), w$ is a prefix of a play, and for all $0 \leq i \leq n.\ \gamma(s_i) = o_i\}$. The belief-updates associated with finite-prefixes are as follows: for prefixes w and $w' = w\cdot a\cdot s$ the belief update is defined inductively $\mathcal{B}(\gamma(w')) = \left(\bigcup_{s_1\in\mathcal{B}(\gamma(w))} \mathrm{Supp}(\delta(s_1, a))\right) \cap \gamma^{-1}(\gamma(s))$.

Strategies. A *strategy (or a policy)* is a recipe to extend prefixes of plays and is a function $\sigma : (S \cdot A)^* \cdot S \to \mathcal{D}(A)$ that given a finite history (i.e., a finite prefix of a play) selects a probability distribution over the actions. Since we consider POMDPs, strategies are *observation-based*, i.e., for all histories $w = (s_0, a_0, s_1, a_1, \ldots, a_{n-1}, s_n)$ and $w' = (s'_0, a_0, s'_1, a_1, \ldots, a_{n-1}, s'_n)$ such that for all $0 \leq i \leq n$ we have $\gamma(s_i) = \gamma(s'_i)$ (i.e., $\gamma(w) = \gamma(w')$), we must have $\sigma(w) = \sigma(w')$. In other words, if the observation sequence is the same, then the strategy cannot distinguish between the prefixes and must play the same. We now present an equivalent definition of observation-based strategies such that the memory of the strategy is explicitly specified, and will be required to present finite-memory strategies.

Definition 2 (Strategies with memory and finite-memory strategies)**.** *A* strategy *with memory is a tuple $\sigma = (\sigma_u, \sigma_n, M, m_0)$ where:(i)* (Memory set). *M is a denumerable set (finite or infinite) of memory elements (or memory states). (ii)* (Action selection function). *The function $\sigma_n : M \to \mathcal{D}(\mathcal{A})$ is the* action selection function *that given the current memory state gives the probability distribution over actions. (iii)* (Memory update function). *The function $\sigma_u : M \times \mathcal{O} \times \mathcal{A} \to \mathcal{D}(M)$ is the* memory update function *that given the current memory state, the current observation and action, updates the memory state probabilistically. (iv)* (Initial memory). *The memory state $m_0 \in M$ is the initial memory state. A strategy is a* finite-memory *strategy if the set M of memory elements is finite. A strategy is* pure (or deterministic) *if the memory update function and the action selection function are deterministic, i.e., $\sigma_u : M \times \mathcal{O} \times \mathcal{A} \to M$ and $\sigma_n : M \to \mathcal{A}$. A strategy is* memoryless (or stationary) *if it is independent of the history but depends only on the current observation, and can be represented as a function $\sigma : \mathcal{O} \to \mathcal{D}(\mathcal{A})$.*

Objectives. An *objective* in a POMDP G is a measurable set $\varphi \subseteq \Omega$ of plays. We first define *limit-average payoff* (aka mean-payoff) function. Given a POMDP we consider a reward function $\mathsf{r} : S \times \mathcal{A} \to [0, 1]$ that maps every state action pair to a real-valued reward in the interval $[0, 1]$. The LimAvg payoff function maps every play to a real-valued reward that is the long-run average of the rewards of the play. Formally, given a play $\rho = (s_0, a_0, s_1, a_1, s_2, a_2, \ldots)$ we have $\mathsf{LimAvg}(\mathsf{r}, \rho) = \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n} \mathsf{r}(s_i, a_i)$. When the reward function r is clear from the context, we drop it for simplicity. For a reward function r, we consider two types of limit-average payoff constraints: (i) *Qualitative constraint.* The *qualitative constraint* limit-average objective $\mathsf{LimAvg}_{=1}$ defines the set of paths such that the limit-average payoff is 1; i.e., $\mathsf{LimAvg}_{=1} = \{\rho \mid \mathsf{LimAvg}(\rho) = 1\}$. (ii) *Quantitative constraints.* Given a threshold $\lambda_1 \in (0,1)$, the *quantitative constraint* limit-average objective $\mathsf{LimAvg}_{>\lambda_1}$ defines the set of paths such that the limit-average payoff is strictly greater than λ_1; i.e., $\mathsf{LimAvg}_{>\lambda_1} = \{\rho \mid \mathsf{LimAvg}(\rho) > \lambda_1\}$.

Probabilistic and almost-sure winning. Given a POMDP, an objective φ, and a class $\mathcal{C}$ of strategies, we say that: (i) a strategy $\sigma \in \mathcal{C}$ is *almost-sure winning* if $\mathbb{P}^\sigma(\varphi) = 1$; (ii) a strategy $\sigma \in \mathcal{C}$ is *probabilistic winning*, for a threshold $\lambda_2 \in (0,1)$, if $\mathbb{P}^\sigma(\varphi) \geq \lambda_2$.

Theorem 1 (Results for PFA (probabilistic automata over finite words) [21]). *The following assertions hold for the class $\mathcal{C}$ of all infinite-memory as well as finite-memory strategies: (1) the probabilistic winning problem is undecidable for PFA; and (2) the almost-sure winning problem is PSPACE-complete for PFA.*

Since PFA are a special case of POMDPs, the undecidability of the probabilistic winning problem for PFA implies the undecidability of the probabilistic winning problem for POMDPs with both qualitative and quantitative constraint limit-average objectives. The almost-sure winning problem is PSPACE-complete for PFAs, and we study the complexity of the almost-sure winning problem for POMDPs with both qualitative and quantitative constraint limit-average objectives, under infinite-memory and finite-memory strategies.

Basic properties of Markov Chains. Since our proofs will use results of Markov chains, we start with some basic results related to Markov chains.

Markov chains and recurrent classes. A Markov chain $\overline{G} = (\overline{S}, \overline{\delta})$ consists of a finite set $\overline{S}$ of states and a probabilistic transition function $\overline{\delta} : \overline{S} \to \mathcal{D}(\overline{S})$. Given the Markov chain, we consider the directed graph $(\overline{S}, \overline{E})$ where $\overline{E} = \{(\overline{s}, \overline{s}') \mid \delta(\overline{s}' \mid \overline{s}) > 0\}$. A *recurrent class* $\overline{C} \subseteq \overline{S}$ of the Markov chain is a bottom strongly connected component (scc) in the graph $(\overline{S}, \overline{E})$ (a bottom scc is an scc with no edges out of the scc). We denote by $\mathsf{Rec}(\overline{G})$ the set of recurrent classes of the Markov chain, i.e., $\mathsf{Rec}(\overline{G}) = \{\overline{C} \mid \overline{C} \text{ is a recurrent class}\}$. Given a state $\overline{s}$ and a set $\overline{U}$ of states, we say that $\overline{U}$ is reachable from $\overline{s}$ if there is a path from $\overline{s}$ to some state in $\overline{U}$ in the graph $(\overline{S}, \overline{E})$. Given a state $\overline{s}$ of the Markov chain we denote by $\mathsf{Rec}(\overline{G})(\overline{s}) \subseteq \mathsf{Rec}(\overline{G})$ the subset of the recurrent classes reachable from $\overline{s}$ in $\overline{G}$. A state is *recurrent* if it belongs to a recurrent class. The following standard properties of reachability and the recurrent classes will be used in our proofs:

- *Property 1.* (a) For a set $\overline{T} \subseteq \overline{S}$, if for all states $\overline{s} \in \overline{S}$ there is a path to $\overline{T}$ (i.e., for all states there is a positive probability to reach $\overline{T}$), then from all states the set $\overline{T}$ is reached with probability 1. (b) For all states $\overline{s}$, if the Markov chain starts at $\overline{s}$, then the set $\overline{\mathsf{C}} = \bigcup_{\overline{C} \in \mathsf{Rec}(\overline{G})(\overline{s})} \overline{C}$ is reached with probability 1, i.e., the set of recurrent classes is reached with probability 1.
- *Property 2.* For a recurrent class $\overline{C}$, for all states $\overline{s} \in \overline{C}$, if the Markov chain starts at $\overline{s}$, then for all states $\overline{t} \in \overline{C}$ the state $\overline{t}$ is visited infinitely often with probability 1, and is visited with positive average frequency (i.e., positive limit-average frequency) with probability 1.

Lemma 1 is a consequence of the above properties.

Lemma 1. *Let $\overline{G} = (\overline{S}, \overline{\delta})$ be a Markov chain with a reward function $\mathsf{r} : \overline{S} \to [0,1]$, and $\overline{s} \in \overline{S}$ a state of the Markov chain. The state $\overline{s}$ is almost-sure winning for the objective $\mathsf{LimAvg}_{=1}$ iff for all recurrent classes $\overline{C} \in \mathsf{Rec}(\overline{G})(\overline{s})$ and for all states $\overline{s}_1 \in \overline{C}$ we have $\mathsf{r}(\overline{s}_1) = 1$.*

Markov chains under finite memory strategies. We now define Markov chains obtained by fixing a finite-memory strategy in a POMDP G. A finite-memory strategy $\sigma = (\sigma_u, \sigma_n, M, m_0)$ induces a Markov chain $(S \times M, \delta_\sigma)$, denoted $G{\restriction}_\sigma$, with the probabilistic transition function $\delta_\sigma : S \times M \to \mathcal{D}(S \times M)$: given $s, s' \in S$ and $m, m' \in M$, the transition $\delta_\sigma\big((s', m') \mid (s, m)\big)$ is the probability to go from state (s, m) to state (s', m') in one step under the strategy σ. The probability of transition can be decomposed as follows: (i) First an action $a \in \mathcal{A}$ is sampled according to the distribution $\sigma_n(m)$; (ii) then the next state s' is sampled according to the distribution $\delta(s, a)$; and (iii) finally the new memory m' is sampled according to the distribution $\sigma_u(m, \gamma(s'), a)$ (i.e., the new memory is sampled according σ_u given the old memory, new observation and the action). More formally, we have: $\delta_\sigma\big((s', m') \mid (s, m)\big) = \sum_{a \in \mathcal{A}} \sigma_n(m)(a) \cdot \delta(s, a)(s') \cdot \sigma_u(m, \gamma(s'), a)(m')$.

3 Finite-memory strategies with Qualitative Constraint

In this section we show the following three results for finite-memory strategies: (i) in POMDPs with $\mathsf{LimAvg}_{=1}$ objectives belief-based strategies are not sufficient for almost-sure winning; (ii) an exponential upper bound on the memory required by an almost-sure winning strategy for $\mathsf{LimAvg}_{=1}$ objectives; and (iii) the decision problem is EXPTIME-complete.

Belief is not sufficient. We now show with an example that there exist POMDPs with $\mathsf{LimAvg}_{=1}$ objectives, where finite-memory randomized almost-sure winning strategies exist, but there exists no belief-based randomized almost-sure winning strategy (a belief-based strategy only uses memory that relies on the subset construction where the subset denotes the possible current states called belief). We will present the counter-example even for POMDPs with restricted reward function r assigning only Boolean rewards 0 and 1 to the states (does not depend on the action).

Example 1. *We consider a POMDP with state space $\{s_0, X, X', Y, Y', Z, Z'\}$ and action set $\{a, b\}$, and let $U = \{X, X', Y, Y', Z, Z'\}$. From the initial state s_0 all the other states are reached with uniform probability in one-step, i.e., for all $s' \in U = \{X, X', Y, Y', Z, Z'\}$ we have $\delta(s_0, a)(s') = \delta(s_0, b)(s') = \frac{1}{6}$. The transitions from the other states are shown in Figure 1. All states in U have the same observation. The reward function* r *assigns the reward 1 to states X, X', Z, Z' and 0 to states Y and Y'. The belief initially after one-step is the set $U = \{X, X', Y, Y', Z, Z'\}$ since from s_0 all of them are reached with positive probability. The belief is always the set U since every state has an input edge for every action, i.e., if the current belief is U (the set of states that the POMDP is currently in with positive probability is U), then irrespective of whether a or b is chosen all states of U are reached with positive probability and thus the belief is again U. There are three belief-based strategies: (i) σ_1 that plays always a; (ii) σ_2 that plays always b; or (iii) σ_3 that plays both a and b with positive probability. The Markov chains $G\!\restriction_{\sigma_1}$ and $G\!\restriction_{\sigma_2}$ are also shown in Figure 1, and the graph of $G\!\restriction_{\sigma_3}$ is the same as the POMDP G (with edge labels removed). For all the three strategies, the Markov chains contain the whole set U as the reachable recurrent class, and it follows by Lemma 1 that none of the belief-based strategies σ_1, σ_2 or σ_3 are almost-sure winning for the* $\mathsf{LimAvg}_{=1}$ *objective. The strategy σ_4 that plays action a and b alternately gives rise to a Markov chain where the recurrent classes do not intersect with Y or Y', and is a finite-memory almost-sure winning strategy for the* $\mathsf{LimAvg}_{=1}$ *objective.* □

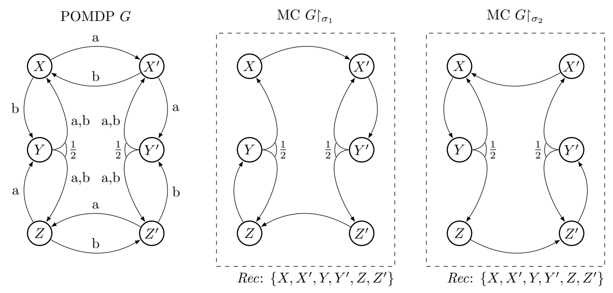

POMDP G
MC $G\!\restriction_{\sigma_1}$
MC $G\!\restriction_{\sigma_2}$
X
X'
Y
Y'
Z
Z'
a
b
a,b
$\frac{1}{2}$
Rec: $\{X, X', Y, Y', Z, Z'\}$
Rec: $\{X, X', Y, Y', Z, Z'\}$

Figure 1: Belief is not sufficient

3.1 Strategy complexity

For the rest of the subsection we fix a finite-memory almost-sure winning strategy $\sigma = (\sigma_u, \sigma_n, M, m_0)$ on the POMDP $G = (S, \mathcal{A}, \delta, \mathcal{O}, \gamma, s_0)$ with a reward function r for the objective $\mathsf{LimAvg}_{=1}$. Our goal is to construct an almost-sure winning strategy for the $\mathsf{LimAvg}_{=1}$ objective with memory size at most $\mathsf{Mem}^* = 2^{3\cdot|S|} \cdot 2^{|\mathcal{A}|}$. We start with a few definitions associated with strategy σ. For $m \in M$:

- The function $\mathsf{RecFun}_\sigma(m) : S \to \{0, 1\}$ is such that $\mathsf{RecFun}_\sigma(m)(s)$ is 1 iff the state (s, m) is recurrent in the Markov chain $G\!\restriction_\sigma$ and 0 otherwise.
- The function $\mathsf{AWFun}_\sigma(m) : S \to \{0, 1\}$ is such that $\mathsf{AWFun}_\sigma(m)(s)$ is 1 iff the state (s, m) is almost-sure winning for the $\mathsf{LimAvg}_{=1}$ objective in the Markov chain $G\!\restriction_\sigma$ and 0 otherwise.
- We also consider $\mathsf{Act}_\sigma(m) = \mathrm{Supp}(\sigma_n(m))$ that for every memory element gives the support of the probability distribution over actions played at m.

Remark 1. *Let (s', m') be reachable from (s, m) in $G\!\restriction_\sigma$. If the state (s, m) is almost-sure winning for the* $\mathsf{LimAvg}_{=1}$ *objective, then the state (s', m') is also almost-sure winning for the* $\mathsf{LimAvg}_{=1}$ *objective.*

Collapsed graph of σ. Given the strategy σ we define the notion of a *collapsed graph* $\mathsf{CoGr}(\sigma) = (V, E)$. The states of the graph are elements from the set $V = \{(Y, \mathsf{AWFun}_\sigma(m), \mathsf{RecFun}_\sigma(m), \mathsf{Act}_\sigma(m)) \mid Y \subseteq S \text{ and } m \in M\}$ and the initial state is $(\{s_0\}, \mathsf{AWFun}_\sigma(m_0), \mathsf{RecFun}_\sigma(m_0), \mathsf{Act}_\sigma(m_0))$. The edges in E are labeled by actions in $\mathcal{A}$. Intuitively, the action labeled edges of the graph depict the updates of the belief and the functions upon a particular action. Formally, there is an edge $(Y, \mathsf{AWFun}_\sigma(m), \mathsf{RecFun}_\sigma(m), \mathsf{Act}_\sigma(m)) \xrightarrow{a} (Y', \mathsf{AWFun}_\sigma(m'), \mathsf{RecFun}_\sigma(m'), \mathsf{Act}_\sigma(m'))$ in the collapsed graph $\mathsf{CoGr}(\sigma)$ iff there exists an observation $o \in \mathcal{O}$ such that (i) the action $a \in \mathsf{Act}_\sigma(m)$; (ii) the set Y' is non-empty and it is the belief update from Y, under action a and the observation o, i.e., $Y' = \bigcup_{s \in Y} \mathrm{Supp}(\delta(s, a)) \cap \gamma^{-1}(o)$; and (iii) $m' \in \mathrm{Supp}(\sigma_u(m, o, a))$. Note that the number of states in the graph is bounded by $|V| \leq 2^{3\cdot|S|} \cdot 2^{|\mathcal{A}|} = \mathsf{Mem}^*$.

We now define the collapsed strategy for σ. Intuitively we collapse memory elements of σ whenever they agree on all the RecFun, AWFun, and Act functions. The collapsed strategy plays uniformly all the actions from the set given by Act in the collapsed state.

Collapsed strategy. We now construct the *collapsed strategy* $\sigma' = (\sigma'_u, \sigma'_n, M', m'_0)$ of σ based on the collapsed graph $\mathsf{CoGr}(\sigma) = (V, E)$. We will refer to this construction by $\sigma' = \mathsf{CoSt}(\sigma)$.

- The memory set M' are the vertices of the collapsed graph $\mathsf{CoGr}(\sigma) = (V, E)$, i.e., $M' = V =$

$\{(Y, \mathsf{AWFun}_\sigma(m), \mathsf{RecFun}_\sigma(m), \mathsf{Act}_\sigma(m)) \mid Y \subseteq S \text{ and } m \in M\}$. Note that $|M'| \leq \mathsf{Mem}^*$.
- The initial memory is $m'_0 = (\{s_0\}, \mathsf{AWFun}_\sigma(m_0), \mathsf{RecFun}_\sigma(m_0), \mathsf{Act}_\sigma(m_0))$.
- The next action function given a memory $(Y, W, R, A) \in M'$ is the uniform distribution over the set of actions $\{a \mid \exists (Y', W', R', A') \in M' \text{ and } (Y, W, R, A) \xrightarrow{a} (Y', W', R', A') \in E\}$, where E are the edges of the collapsed graph.
- The memory update function $\sigma'_u((Y, W, R, A), o, a)$ given a memory element $(Y, W, R, A) \in M'$, $a \in \mathcal{A}$, and $o \in \mathcal{O}$ is the uniform distribution over the set of states $\{(Y', W', R', A') \mid (Y, W, R, A) \xrightarrow{a} (Y', W', R', A') \in E \text{ and } Y' \subseteq \gamma^{-1}(o)\}$.

Random variable notation. For all $n \geq 0$ we write $X_n, Y_n, W_n, R_n, A_n, L_n$ for the random variables that correspond to the projection of the n^{th} state of the Markov chain $G\restriction_{\sigma'}$ on the S component, the belief $\mathcal{P}(S)$ component, the AWFun_σ component, the RecFun_σ component, the Act_σ component, and the n^{th} action, respectively.

Run of the Markov chain $G\restriction_{\sigma'}$. A *run* of the Markov chain $G\restriction_{\sigma'}$ is an infinite sequence

$$(X_0, Y_0, W_0, R_0, A_0) \xrightarrow{L_0} (X_1, Y_1, W_1, R_1, A_1) \xrightarrow{L_1} \cdots$$

such that each finite prefix of the run is generated with positive probability on the Markov chain, i.e., for all $i \geq 0$, we have (i) $L_i \in \mathrm{Supp}(\sigma'_n(Y_i, W_i, R_i, A_i))$; (ii) $X_{i+1} \in \mathrm{Supp}(\delta(X_i, L_i))$; and (iii) $(Y_{i+1}, W_{i+1}, R_{i+1}, A_{i+1}) \in \mathrm{Supp}(\sigma'_u((Y_i, W_i, R_i, A_i), \gamma(X_{i+1}), L_i))$. In the following lemma we establish important properties of the Markov chain $G\restriction_{\sigma'}$ that are essential for our proof.

Lemma 2. *Let* $(X_0, Y_0, W_0, R_0, A_0) \xrightarrow{L_0} (X_1, Y_1, W_1, R_1, A_1) \xrightarrow{L_1} \cdots$ *be a run of the Markov chain* $G\restriction_{\sigma'}$*, then the following assertions hold for all* $i \geq 0$*: 1.* $X_{i+1} \in \mathrm{Supp}(\delta(X_i, L_i)) \cap Y_{i+1}$*; 2.* $(Y_i, W_i, R_i, A_i) \xrightarrow{L_i} (Y_{i+1}, W_{i+1}, R_{i+1}, A_{i+1})$ *is an edge in the collapsed graph* $\mathsf{CoGr}(\sigma)$*; 3. if* $W_i(X_i) = 1$*, then* $W_{i+1}(X_{i+1}) = 1$*; 4. if* $R_i(X_i) = 1$*, then* $R_{i+1}(X_{i+1}) = 1$*; and 5. if* $W_i(X_i) = 1$ *and* $R_i(X_i) = 1$*, then* $\mathsf{r}(X_i, L_i) = 1$*.*

Proof. We present the proof of the fifth point, and the other points are straight-forward. For the fifth point consider that $W_i(X_i) = 1$ and $R_i(X_i) = 1$. Then there exists a memory $m \in M$ such that (i) $\mathsf{AWFun}_\sigma(m) = W_i$, and (ii) $\mathsf{RecFun}_\sigma(m) = R_i$. Moreover, the state (X_i, m) is a recurrent (since $R_i(X_i) = 1$) and almost-sure winning state (since $W_i(X_i) = 1$) in the Markov chain $G\restriction_\sigma$. As $L_i \in \mathsf{Act}_\sigma(m)$ it follows that $L_i \in \mathrm{Supp}(\sigma_n(m))$, i.e., the action L_i is played with positive probability in state X_i given memory m, and (X_i, m) is in an almost-sure winning recurrent class. By Lemma 1 it follows that the reward $\mathsf{r}(X_i, L_i)$ must be 1. The desired result follows. ☐

We now introduce the final notion of a collapsed-recurrent state that is required to complete the proof. A state (X, Y, W, R, A) of the Markov chain $G\restriction_{\sigma'}$ is collapsed-recurrent, if for all memory elements $m \in M$ that were merged to the memory element (Y, W, R, A), the state (X, m) of the Markov chain $G\restriction_\sigma$ is recurrent. It will turn out that every recurrent state of the Markov chain $G\restriction_\sigma$ is also collapsed-recurrent.

Definition 3. *A state* (X, Y, W, R, A) *of the Markov chain* $G\restriction_{\sigma'}$ *is called collapsed-recurrent iff* $R(X) = 1$*.*

Note that due to point 4 of Lemma 2 all the states reachable from a collapsed-recurrent state are also collapsed-recurrent. In the following lemma we show that the set of collapsed-recurrent states is reached with probability 1; and the key fact we show is that from every state in $G\restriction_{\sigma'}$ a collapsed-recurrent state is reached with positive probability, and then use Property 1 (a) of Markov chains to establish the lemma.

Lemma 3. *With probability 1 a run of the Markov chain* $G\restriction_{\sigma'}$ *reaches a collapsed-recurrent state.*

Lemma 4. *The collapsed strategy* σ' *is a finite-memory almost-sure winning strategy for the* $\mathsf{LimAvg}_{=1}$ *objective on the POMDP* G *with the reward function* r*.*

Proof. The initial state of the Markov chain $G\restriction_{\sigma'}$ is $(\{s_0\}, \mathsf{AWFun}_\sigma(m_0), \mathsf{RecFun}_\sigma(m_0), \mathsf{Act}_\sigma(m_0))$ and as the strategy σ is an almost-sure winning strategy we have that $\mathsf{AWFun}_\sigma(m_0)(s_0) = 1$. It follows from the third point of Lemma 2 that every reachable state (X, Y, W, R, A) in the Markov chain $G\restriction_{\sigma'}$ satisfies that $W(X) = 1$. From the initial state a collapsed-recurrent state is reached with probability 1. It follows that all the recurrent states in the Markov chain $G\restriction_{\sigma'}$ are also collapsed-recurrent states. As in all reachable states (X, Y, W, R, A) we have $W(X) = 1$, by the fifth point of Lemma 2 it follows that every action L played in a collapsed-recurrent state (X, Y, W, R, A) satisfies that the reward $\mathsf{r}(X, L) = 1$. As this true for every reachable recurrent class, the fact that the collapsed strategy is an almost-sure winning strategy for $\mathsf{LimAvg}_{=1}$ objective follows from Lemma 1. ☐

Theorem 2 (Strategy complexity)**.** *The following assertions hold: (1) If there exists a finite-memory almost-sure winning strategy in the POMDP* $G = (S, \mathcal{A}, \delta, \mathcal{O}, \gamma, s_0)$ *with reward function* r *for the* $\mathsf{LimAvg}_{=1}$ *objective, then there exists a finite-memory* *almost-sure winning strategy with memory size at most $2^{3\cdot|S|+|\mathcal{A}|}$. (2) Finite-memory almost-sure winning strategies for $\mathsf{LimAvg}_{=1}$ objectives in POMDPs in general require exponential memory and belief-based strategies are not sufficient.*

Proof. The first point follows from Lemma 4 and the fact that the size of the memory set of the collapsed strategy σ' of any finite-memory strategy σ (which is the size of the vertex set of the collapsed graph of σ) is bounded by $2^{3\cdot|S|+|\mathcal{A}|}$. $\square$

3.2 Computational complexity

A naive double-exponential time algorithm would be to enumerate all finite-memory strategies with memory bounded by $2^{3\cdot|S|+|\mathcal{A}|}$ (by Theorem 2). Our improved exponential-time algorithm consists of two steps: (i) first it constructs a special type of a *belief-observation* POMDP $\overline{G}$ from a POMDP G (and $\overline{G}$ is exponential in G); and we show that there exists a finite-memory almost-sure winning strategy for the objective $\mathsf{LimAvg}_{=1}$ in G iff there exists a randomized memoryless almost-sure winning strategy in $\overline{G}$ for the objective $\mathsf{LimAvg}_{=1}$; and (ii) then we show how to determine whether there exists a randomized memoryless almost-sure winning strategy in $\overline{G}$ for the $\mathsf{LimAvg}_{=1}$ objective in polynomial time with respect to the size of $\overline{G}$. For a belief-observation POMDP the current belief is always the set of states with current observation.

Definition 4. *A POMDP $\overline{G} = (\overline{S}, \overline{\mathcal{A}}, \overline{\delta}, \overline{\mathcal{O}}, \overline{\gamma}, \overline{s}_0)$ is a* belief-observation *POMDP iff for every finite prefix $\overline{w} = (\overline{s}_0, \overline{a}_0, \overline{s}_1, \overline{a}_1, \ldots, \overline{a}_{n-1}, \overline{s}_n)$ the belief associated with the observation sequence $\overline{\rho} = \overline{\gamma}(\overline{w})$ is the set of states with the last observation $\gamma(\overline{s}_n)$ of the observation sequence $\overline{\rho}$, i.e., $\mathcal{B}(\overline{\rho}) = \overline{\gamma}^{-1}(\overline{\gamma}(\overline{s}_n))$.*

Construction of the belief-observation POMDP. Intuitively, the construction of $\overline{G}$ from G will proceed as follows: if there exists an almost-sure winning finite-memory strategy, then there exists an almost-sure winning collapsed strategy with memory bounded by $2^{3\cdot|S|+|\mathcal{A}|}$. This allows us to consider the memory elements $M = 2^S \times \{0,1\}^{|S|} \times \{0,1\}^{|S|} \times 2^{\mathcal{A}}$; and intuitively construct the product of the memory M with the POMDP G. The POMDP $\overline{G}$ is constructed such that it allows all possible ways the collapsed strategy of a finite-memory almost-sure winning strategy could play. The reward function $\overline{\mathsf{r}}$ in $\overline{G}$ is obtained from the reward function r in G. In the POMDP $\overline{G}$ the belief is already included in the state space itself of the POMDP, and the belief represents exactly the set of states in which the POMDP can be with positive probability. Therefore, the POMDP $\overline{G}$ is a belief-observation POMDP. Since possible memory states of collapsed strategies are part of state space, we only need to consider memoryless strategies in $\overline{G}$.

Lemma 5. *The POMDP $\overline{G}$ is a belief-observation POMDP, such that there exists a finite-memory almost-sure winning strategy for the $\mathsf{LimAvg}_{=1}$ objective with the reward function r in the POMDP G iff there exists a memoryless almost-sure winning strategy for the $\mathsf{LimAvg}_{=1}$ objective with the reward function $\overline{\mathsf{r}}$ in the POMDP $\overline{G}$.*

Almost-sure winning observations. Given a POMDP $\overline{G} = (\overline{S}, \overline{\mathcal{A}}, \overline{\delta}, \overline{\mathcal{O}}, \overline{\gamma})$ and an objective ψ, let $\mathsf{Almost}_{\mathcal{M}}(\psi)$ denote the set of observations $\overline{o} \in \overline{\mathcal{O}}$, such that there exists a memoryless almost-sure winning strategy to ensure ψ from every state $\overline{s} \in \overline{\gamma}^{-1}(\overline{o})$.

Almost-sure winning for $\mathsf{LimAvg}_{=1}$ objectives. Our goal is to compute the set $\mathsf{Almost}_{\mathcal{M}}(\mathsf{LimAvg}_{=1})$ given the belief-observation POMDP $\overline{G}$ (of our construction of G with product with M). Let $F \subseteq \overline{S}$ be the set of states of $\overline{G}$ where some actions can be played consistently with the collapsed strategy of any finite-memory almost-sure winning strategy. Let $\widetilde{G}$ denote the POMDP $\overline{G}$ restricted to F. We define a subset of states of the belief-observation POMDP $\widetilde{G}$ that intuitively correspond to winning collapsed-recurrent states (wcs), i.e., $\widetilde{S}_{wcs} = \{(s,(Y,W,R,A)) \mid W(s) = 1, R(s) = 1\}$. Then, we compute the set of observations $\widetilde{\mathsf{AW}}$ that can ensure to reach S_{wcs} almost-surely in the the POMDP $\widetilde{G}$. We show that the set of observations $\widetilde{\mathsf{AW}}$ is equal to the set of observations $\mathsf{Almost}_{\mathcal{M}}(\mathsf{LimAvg}_{=1})$ in the POMDP $\overline{G}$. Thus the computation reduces to computation of almost-sure states for reachability objectives. Finally we show that almost-sure reachability set can be computed in quadratic time for belief-observation POMDPs. The quadratic time algorithm is obtained as follows: we show almost-sure winning observations to ensure to reach a target set $\overline{T}$ with probability 1 is the greatest fixpoint of a set $\overline{Y}$ of observations such that playing all actions uniformly that ensures $\overline{Y}$ is not left, ensures to reach $\overline{T}$ almost-surely. This characterization gives a nested iterative algorithm that is quadratic time.

Lemma 6. $\widetilde{\mathsf{AW}} = \mathsf{Almost}_{\mathcal{M}}(\mathsf{LimAvg}_{=1})$*; and* $\widetilde{\mathsf{AW}}$ *can be computed in quadratic time in the size of $\overline{G}$.*

The EXPTIME-completeness. In this section we first showed that given a POMDP G with a $\mathsf{LimAvg}_{=1}$ objective we can construct an exponential size belief-observation POMDP $\overline{G}$ and the computation of the almost-sure winning set for $\mathsf{LimAvg}_{=1}$ objectives is reduced to the computation of the almost-sure winning set for reachability objectives, which we solve in quadratic time in $\overline{G}$. This gives us an exponential-time algorithm to decide (and construct if one ex-

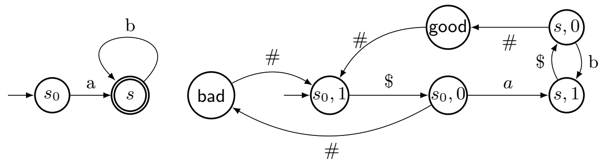


Figure 2: PFA P to a POMDP G

ists) the existence of finite-memory almost-sure winning strategies in POMDPs with $\mathsf{LimAvg}_{=1}$ objectives. The EXPTIME-hardness for almost-sure winning can be obtained easily from the result of Reif for two-player partial-observation games with safety objectives [25].

Theorem 3. *The following assertions hold: (1) Given a POMDP G with $|S|$ states, $|\mathcal{A}|$ actions, and a $\mathsf{LimAvg}_{=1}$ objective, the existence (and construction if one exists) of a finite-memory almost-sure winning strategy can be achieved in $2^{O(|S|+|\mathcal{A}|)}$ time. (2) The decision problem of given a POMDP and a $\mathsf{LimAvg}_{=1}$ objective whether there exists a finite-memory almost-sure winning strategy is EXPTIME-complete.*

Remark 2. *We considered observation function that assigns an observation to every state. In general the observation function $\gamma : S \to 2^{\mathcal{O}} \setminus \emptyset$ may assign multiple observations to a single state. In that case we consider the set of observations as $\mathcal{O}' = 2^{\mathcal{O}} \setminus \emptyset$ and consider the mapping that assigns to every state an observation from $\mathcal{O}'$ and then apply our results.*

4 Finite-memory strategies with Quantitative Constraint

We will show that the problem of deciding whether there exists a finite-memory (as well as an infinite-memory) almost-sure winning strategy for the objective $\mathsf{LimAvg}_{>\frac{1}{2}}$ is undecidable. We present a reduction from the standard undecidable problem for probabilistic finite automata (PFA). A PFA $\mathsf{P} = (S, \mathcal{A}, \delta, F, s_0)$ is a special case of a POMDP $G = (S, \mathcal{A}, \delta, \mathcal{O}, \gamma, s_0)$ with a single observation $\mathcal{O} = \{o\}$ such that for all states $s \in S$ we have $\gamma(s) = o$. Moreover, the PFA proceeds for only finitely many steps, and has a set F of desired final states. The *strict emptiness problem* asks for the existence of a strategy w (a finite word over the alphabet $\mathcal{A}$) such that the measure of the runs ending in the desired final states F is strictly greater than $\frac{1}{2}$; and the strict emptiness problem for PFA is undecidable [21].

Reduction. Given a PFA $\mathsf{P} = (S, \mathcal{A}, \delta, F, s_0)$ we construct a POMDP $G = (S', \mathcal{A}', \delta', \mathcal{O}, \gamma, s_0')$ with a Boolean reward function r such that there exists a word $w \in \mathcal{A}^*$ accepted with probability strictly greater than $\frac{1}{2}$ in P iff there exists a finite-memory almost-sure winning strategy in G for the objective $\mathsf{LimAvg}_{>\frac{1}{2}}$. Intuitively, the construction of the POMDP G is as follows: for every state $s \in S$ of P we construct a pair of states $(s, 1)$ and $(s, 0)$ in S' with the property that $(s, 0)$ can only be reached with a new action \$ (not in $\mathcal{A}$) played in state $(s, 1)$. The transition function δ' from the state $(s, 0)$ mimics the transition function δ, i.e., $\delta'((s, 0), a)((s', 1)) = \delta(s, a)(s')$. The reward r of $(s, 1)$ (resp. $(s, 0)$) is 1 (resp. 0), ensuring the average of the pair to be $\frac{1}{2}$. We add a new available action # that when played in a final state reaches a state $\mathsf{good} \in S'$ with reward 1, and when played in a non-final state reaches a state $\mathsf{bad} \in S'$ with reward 0, and for states good and bad given action # the next state is the initial state. An illustration of the construction on an example is depicted on Figure 2. Whenever an action is played in a state where it is not available, the POMDP reaches a loosing absorbing state, i.e., an absorbing state with reward 0, and for brevity we omit transitions to the loosing absorbing state. We present key proof ideas to establish the correctness:

(Strict emptiness implies almost-sure $\mathsf{LimAvg}_{>\frac{1}{2}}$). Let $w \in \mathcal{A}^*$ be a word accepted in P with probability $\mu > \frac{1}{2}$ and let the length of the word be $|w| = n$. We construct a pure finite-memory almost-sure winning strategy for the objective $\mathsf{LimAvg}_{>\frac{1}{2}}$ in the POMDP G as follows: We denote by $w[i]$ the i^{th} action in the word w. The finite-memory strategy we construct is specified as an ultimately periodic word $(\$\, w[1]\, \$\, w[2] \ldots \$ w[n]\, \#\#)^{\omega}$. Observe that by the construction of the POMDP G, the sequence of rewards (that appear on the transitions) is $(10)^n$ followed by (i) 1 with probability μ (when F is reached), and (ii) 0 otherwise; and the whole sequence is repeated ad infinitum. Then using the Strong Law of Large Numbers (SLLN) [6, Theorem 7.1, page 56] we show that with probability 1 the objective $\mathsf{LimAvg}_{>\frac{1}{2}}$ is satisfied.

(Almost-sure $\mathsf{LimAvg}_{>\frac{1}{2}}$ implies strict emptiness). Conversely, if there is a pure finite-memory strategy σ to ensure the objective $\mathsf{LimAvg}_{>\frac{1}{2}}$ in the POMDP, then the strategy σ can be viewed as an ultimately periodic infinite word of the form $u \cdot v^{\omega}$, where u, v are finite words from $\mathcal{A}'$. Note that v must contain the subsequence ##, as otherwise the LimAvg payoff would be only $\frac{1}{2}$. Similarly, before every letter $a \in \mathcal{A}$ in the words u, v, the strategy must necessarily play the \$ action, as otherwise the loosing absorbing state is reached. Again using SLLN we show that from the word v we can extract a word w that is accepted in the PFA with probability strictly greater than $\frac{1}{2}$. Finally, we show that if there is randomized (possibly infinite-memory strategy) to ensure the objective $\mathsf{LimAvg}_{>\frac{1}{2}}$ in the POMDP, then there is a pure finite-memory strategy as well (the technical proof uses Fa-

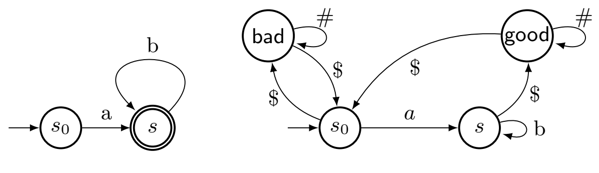


Figure 3: PFA P to a POMDP G

tou's lemma [6]).

Theorem 4. *The problem whether there exists a finite (or infinite-memory) almost-sure winning strategy in a POMDP for the objective* $\mathsf{LimAvg}_{>\frac{1}{2}}$ *is undecidable.*

5 Infinite-memory strategies with Qualitative Constraint

In this section we show that the problem of deciding the existence of infinite-memory almost-sure winning strategies in POMDPs with $\mathsf{LimAvg}_{=1}$ objectives is undecidable. We prove this fact by a reduction from the *value 1 problem* in PFA, which is undecidable [8]. The value 1 problem given a PFA P asks whether for every $\epsilon > 0$ there exists a finite word w such that the word is accepted in P with probability at least $1 - \epsilon$ (i.e., the limit of the acceptance probabilities is 1).

Reduction. Given a PFA $\mathsf{P} = (S, \mathcal{A}, \delta, F, s_0)$, we construct a POMDP $G' = (S', \mathcal{A}', \delta', \mathcal{O}', \gamma', s'_0)$ with a reward function r', such that P satisfies the value 1 problem iff there exists an infinite-memory almost-sure winning strategy in G' for the objective $\mathsf{LimAvg}_{=1}$. Intuitively, the construction adds two additional states good and bad. We add an edge from every state of the PFA under a new action \$, this edge leads to the state good when played in a final state, and to the state bad otherwise. In the states good and bad we add self-loops under a new action #. The action \$ in the states good or bad leads back to the initial state. An example of the construction is illustrated with Figure 3. All the states belong to a single observation, and we will use Boolean reward function on states. The reward for all states except the newly added state good is 0, and the reward for the state good is 1.

The key proof ideas for correctness are as follows:

(Value 1 implies almost-sure $\mathsf{LimAvg}_{=1}$*).* If P satisfies the value 1 problem, then there exists a sequence of finite words $(w_i)_{i\geq 1}$, such that each w_i is accepted in P with probability at least $1-\frac{1}{2^{i+1}}$. We construct an infinite word $w_1 \cdot \$ \cdot \#^{n_1} \cdot w_2 \cdot \$ \cdot \#^{n_2} \cdots$, where each $n_i \in \mathbb{N}$ is a natural number that satisfies the following condition: let $k_i = |w_{i+1} \cdot \$| + \sum_{j=1}^{i}(|w_j \cdot \$| + n_j)$ be the length of the word sequence before $\#^{n_{i+1}}$, then we must have $\frac{n_i}{k_i} \geq 1 - \frac{1}{i}$. The construction ensures that if the state bad appears only finitely often with probability 1, then $\mathsf{LimAvg}_{=1}$ is ensured with probability 1. The argument to show that bad is visited infinitely often with probability 0 is as follows. We first upper bound the probability u_{k+1} to visit the state bad at least $k+1$ times, given k visits to state bad. The probability u_{k+1} is at most $\frac{1}{2^{k+1}}(1 + \frac{1}{2} + \frac{1}{4} + \cdots)$. The above bound for u_{k+1} is obtained as follows: following the visit to bad for k times, the words w_j, for $j \geq k$ are played; and hence the probability to reach bad decreases by $\frac{1}{2}$ every time the next word is played; and after k visits the probability is always smaller than $\frac{1}{2^{k+1}}$. Hence the probability to visit bad at least $k+1$ times, given k visits, is at most the sum above, which is $\frac{1}{2^k}$. Let $\mathcal{E}_k$ denote the event that bad is visited at least $k+1$ times given k visits to bad. Then we have $\sum_{k\geq 0} \mathbb{P}(\mathcal{E}_k) \leq \sum_{k\geq 1} \frac{1}{2^k} < \infty$. By Borel-Cantelli lemma [6, Theorem 6.1, page 47] we have that the probability that bad is visited infinitely often is 0.

(Almost-sure $\mathsf{LimAvg}_{=1}$ *implies value 1).* We prove the converse. Consider that the PFA P does not satisfy the value 1 problem, i.e., there exists a constant $c > 0$ such that for all $w \in \mathcal{A}^*$ we have that the probability that w is accepted in P is at most $1 - c < 1$. We will show that there is no almost-sure winning strategy. Assume towards contradiction that there exists an infinite-memory almost-sure winning strategy σ in the POMDP G'; and the infinite word corresponding to σ must play infinitely many sequences of #'s to ensure $\mathsf{LimAvg}_{=1}$. Let X_i be the random variable for the rewards for the i-th sequence of #'s. Then we have that $X_i = 1$ with probability at most $1 - c$ and 0 otherwise. The expected LimAvg payoff is then at most: $\mathbb{E}(\liminf_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n} X_i)$. Since X_i's are non-negative measurable function, by Fatou's lemma [6, Theorem 3.5, page 16]

$$\mathbb{E}(\liminf_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n} X_i) \leq \liminf_{n\to\infty} \mathbb{E}(\frac{1}{n} \sum_{i=0}^{n} X_i) \leq 1 - c.$$

It follows that $\mathbb{E}^\sigma(\mathsf{LimAvg}) \leq 1 - c$. Note that if the strategy σ was almost-sure winning for the objective $\mathsf{LimAvg}_{=1}$ (i.e., $\mathbb{P}^\sigma(\mathsf{LimAvg}_{=1}) = 1$), then the expectation of the LimAvg payoff would also be 1 (i.e., $\mathbb{E}^\sigma(\mathsf{LimAvg}) = 1$). Therefore we have reached a contradiction to the fact that the strategy σ is almost-sure winning for the objective $\mathsf{LimAvg}_{=1}$.

Theorem 5. *The problem whether there exists an infinite-memory almost-sure winning strategy in a POMDP with the objective* $\mathsf{LimAvg}_{=1}$ *is undecidable.*

References

[1] S. B. Andersson and D. Hristu. Symbolic feedback control for navigation. *IEEE Transactions*

on Automatic Control, 51(6):926–937, 2006.

[2] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1023–1023. JOHN WILEY & SONS LTD, 1995.

[3] A. Condon and R. J. Lipton. On the complexity of space bounded interactive proofs. In *FOCS*, pages 462–467, 1989.

[4] K. Culik and J. Kari. Digital images and formal languages. *Handbook of formal languages*, pages 599–616, 1997.

[5] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge Univ. Press, 1998.

[6] R. Durrett. *Probability: Theory and Examples (Second Edition)*. Duxbury Press, 1996.

[7] J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer-Verlag, 1997.

[8] H. Gimbert and Y. Oualhadj. Probabilistic automata on finite words: Decidable and undecidable problems. In *Proc. of ICALP*, LNCS 6199, pages 527–538. Springer, 2010.

[9] E. A. Hansen and R. Zhou. Synthesis of hierarchical finite-state controllers for pomdps. In *ICAPS*, pages 113–122, 2003.

[10] H. Howard. *Dynamic Programming and Markov Processes*. MIT Press, 1960.

[11] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

[12] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *J. of Artif. Intell. Research*, 4:237–285, 1996.

[13] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas. Temporal-logic-based reactive mission and motion planning. *IEEE Transactions on Robotics*, 25(6):1370–1381, 2009.

[14] M. L. Littman, J. Goldsmith, and M. Mundhenk. The computational complexity of probabilistic planning. *J. Artif. Intell. Res. (JAIR)*, 9:1–36, 1998.

[15] M. L. Littman. *Algorithms for Sequential Decision Making*. PhD thesis, Brown University, 1996.

[16] O. Madani, S. Hanks, and A. Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artif. Intell.*, 147(1-2):5–34, 2003.

[17] N. Meuleau, L. Peshkin, K-E. Kim, and L.P. Kaelbling. Learning finite-state controllers for partially observable environments. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, UAI'99, pages 427–436, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[18] N. Meuleau, K.-E. Kim, L. P. Kaelbling, and A. R. Cassandra. Solving pomdps by searching the space of finite policies. In *UAI*, pages 417–426, 1999.

[19] M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.

[20] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12:441–450, 1987.

[21] A. Paz. *Introduction to probabilistic automata (Computer science and applied mathematics)*. Academic Press, 1971.

[22] A. Pogosyants, R. Segala, and N. Lynch. Verification of the randomized consensus algorithm of Aspnes and Herlihy: a case study. *Distributed Computing*, 13(3):155–186, 2000.

[23] M. L. Puterman. *Markov Decision Processes*. John Wiley and Sons, 1994.

[24] M. O. Rabin. Probabilistic automata. *Information and Control*, 6:230–245, 1963.

[25] J. H. Reif. The complexity of two-player games of incomplete information. *Journal of Computer and System Sciences*, 29(2):274–301, 1984.

[26] M. I. A. Stoelinga. Fun with FireWire: Experiments with verifying the IEEE1394 root contention protocol. In *Formal Aspects of Computing*, 2002.

[27] J. D. Williams and S. Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422, 2007.

Parallel Gaussian Process Regression with Low-Rank Covariance Matrix Approximations

Jie Chen[†], Nannan Cao[†], Kian Hsiang Low[†], Ruofei Ouyang[†], Colin Keng-Yan Tan[†], and Patrick Jaillet[§]
Department of Computer Science, National University of Singapore, Republic of Singapore[†]
Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, USA[§]

Abstract

Gaussian processes (GP) are Bayesian non-parametric models that are widely used for probabilistic regression. Unfortunately, it cannot scale well with large data nor perform real-time predictions due to its cubic time cost in the data size. This paper presents two parallel GP regression methods that exploit low-rank covariance matrix approximations for distributing the computational load among parallel machines to achieve time efficiency and scalability. We theoretically guarantee the predictive performances of our proposed parallel GPs to be equivalent to that of some centralized approximate GP regression methods: The computation of their centralized counterparts can be distributed among parallel machines, hence achieving greater time efficiency and scalability. We analytically compare the properties of our parallel GPs such as time, space, and communication complexity. Empirical evaluation on two real-world datasets in a cluster of 20 computing nodes shows that our parallel GPs are significantly more time-efficient and scalable than their centralized counterparts and exact/full GP while achieving predictive performances comparable to full GP.

1 Introduction

Gaussian processes (GP) are Bayesian non-parametric models for performing nonlinear regression, which offer an important advantage of providing fully probabilistic predictive distributions with formal measures of the uncertainty of the predictions. The key limitation hindering the practical use of GP for large data is the high computational cost: It incurs cubic time and quadratic memory in the size of the data. To reduce the computational cost, two classes of approximate GP regression methods have been proposed: (a) Low-rank covariance matrix approximation methods (Quiñonero-Candela and Rasmussen, 2005; Snelson and Ghahramani, 2005; Williams and Seeger, 2000) are especially suitable for modeling smoothly-varying functions with high correlation (i.e., long length-scales) and they utilize all the data for predictions like the exact/full GP; and (b) localized regression methods (e.g., local GPs (Das and Srivastava, 2010; Choudhury *et al.*, 2002; Park *et al.*, 2011) and compactly supported covariance functions (Furrer *et al.*, 2006)) are capable of modeling highly-varying functions with low correlation (i.e., short length-scales) but they use only local data for predictions, hence predicting poorly in input regions with sparse data. Recent approximate GP regression methods of Snelson (2007) and Vanhatalo and Vehtari (2008) have attempted to combine the best of both worlds.

Despite these various efforts to scale up GP, it remains computationally impractical for performing real-time predictions necessary in many time-critical applications and decision support systems (e.g., ocean sensing (Cao *et al.*, 2013; Dolan *et al.*, 2009; Low *et al.*, 2007, 2011, 2012; Podnar *et al.*, 2010), traffic monitoring (Chen *et al.*, 2012; Yu *et al.*, 2012), geographical information systems) that need to process and analyze huge quantities of data collected over short time durations (e.g., in astronomy, internet traffic, meteorology, surveillance). To resolve this, the work in this paper considers exploiting clusters of parallel machines to achieve efficient and scalable predictions in real time. Such an idea of scaling up machine learning techniques (e.g., clustering, support vector machines, graphical models) has recently attracted widespread interest in the machine learning community (Bekkerman *et al.*, 2011). For the case of Gaussian process regression, the local GPs method (Das and Srivastava, 2010; Choudhury *et al.*, 2002) appears most straightforward to be "embarrassingly" parallelized but they suffer from discontinuities in predictions on the boundaries of different local GPs. The work of Park *et al.* (2011) rectifies this problem by imposing continuity constraints along the boundaries in a centralized manner. But, its use is restricted strictly to data with 1- and 2-dimensional input features.

This paper presents two parallel GP regression methods (Sections 3 and 4) that, in particular, exploit low-rank covariance matrix approximations for distributing the computational load among parallel machines to achieve time efficiency and scalability. Different from the above-mentioned parallel local GPs method, our proposed parallel GPs do not suffer from boundary effects, work with multi-dimensional input features, and exploit all the data for predictions but do not incur the cubic time cost of the full/exact GP. The specific contributions of our work include:

- Theoretically guaranteeing the predictive performances of our parallel GPs (i.e., *parallel partially independent conditional* (pPIC) and *parallel incomplete Cholesky factorization* (pICF)-based approximations of GP regression model) to be equivalent to that of some centralized approaches to approximate GP regression (Sections 3 and 4). An important practical implication of these results is that the computation of their centralized counterparts can be distributed among a cluster of parallel machines, hence achieving greater time efficiency and scalability. Furthermore, our parallel GPs inherit an advantage of their centralized counterparts in providing a parameter (i.e., size of support set for pPIC and reduced rank for pICF-based GP) to be adjusted in order to trade off between predictive performance and time efficiency;
- Analytically comparing the properties of our parallel GPs such as time, space, and communication complexity, capability of online learning, and practical implications of the structural assumptions (Section 5);
- Implementing our parallel GPs using the *message passing interface* (MPI) framework to run in a cluster of 20 computing nodes and empirically evaluating their predictive performances, time efficiency, scalability, and speedups on two real-world datasets (Section 6).

2 Gaussian Process Regression

The *Gaussian process* (GP) can be used to perform probabilistic regression as follows: Let $\mathcal{X}$ be a set representing the input domain such that each input $x \in \mathcal{X}$ denotes a d-dimensional feature vector and is associated with a realized output value y_x (random output variable Y_x) if it is observed (unobserved). Let $\{Y_x\}_{x\in\mathcal{X}}$ denote a GP, that is, every finite subset of $\{Y_x\}_{x\in\mathcal{X}}$ follows a multivariate Gaussian distribution (Rasmussen and Williams, 2006). Then, the GP is fully specified by its *prior* mean $\mu_x \triangleq \mathbb{E}[Y_x]$ and covariance $\sigma_{xx'} \triangleq \text{cov}[Y_x, Y_{x'}]$ for all $x, x' \in \mathcal{X}$.

Given that a column vector $y_\mathcal{D}$ of realized outputs is observed for some set $\mathcal{D} \subset \mathcal{X}$ of inputs, the GP can exploit this data $(\mathcal{D}, y_\mathcal{D})$ to provide predictions of the unobserved outputs for any set $\mathcal{U} \subseteq \mathcal{X} \setminus \mathcal{D}$ of inputs and their corresponding predictive uncertainties using the following Gaussian *posterior* mean vector and covariance matrix, respectively:

$$\mu_{\mathcal{U}|\mathcal{D}} \triangleq \mu_\mathcal{U} + \Sigma_{\mathcal{U}\mathcal{D}}\Sigma_{\mathcal{D}\mathcal{D}}^{-1}(y_\mathcal{D} - \mu_\mathcal{D}) \tag{1}$$

$$\Sigma_{\mathcal{U}\mathcal{U}|\mathcal{D}} \triangleq \Sigma_{\mathcal{U}\mathcal{U}} - \Sigma_{\mathcal{U}\mathcal{D}}\Sigma_{\mathcal{D}\mathcal{D}}^{-1}\Sigma_{\mathcal{D}\mathcal{U}} \tag{2}$$

where $\mu_\mathcal{U}$ ($\mu_\mathcal{D}$) is a column vector with mean components μ_x for all $x \in \mathcal{U}$ ($x \in \mathcal{D}$), $\Sigma_{\mathcal{U}\mathcal{D}}$ ($\Sigma_{\mathcal{D}\mathcal{D}}$) is a covariance matrix with covariance components $\sigma_{xx'}$ for all $x \in \mathcal{U}, x' \in \mathcal{D}$ ($x, x' \in \mathcal{D}$), and $\Sigma_{\mathcal{D}\mathcal{U}}$ is the transpose of $\Sigma_{\mathcal{U}\mathcal{D}}$. The uncertainty of predicting the unobserved outputs can be measured using the trace of $\Sigma_{\mathcal{U}\mathcal{U}|\mathcal{D}}$ (2) (i.e., sum of posterior variances $\Sigma_{xx|\mathcal{D}}$ over all $x \in \mathcal{U}$), which is independent of the realized outputs $y_\mathcal{D}$.

3 Parallel Gaussian Process Regression using Support Set

The centralized approach to exact/full GP regression described in Section 2, which we call the *full Gaussian process* (FGP), unfortunately cannot scale well and be performed in real time due to its cubic time complexity in the size $|\mathcal{D}|$ of the data. In this section, we will present a class of parallel Gaussian processes (pPITC and pPIC) that distributes the computational load among parallel machines to achieve efficient and scalable approximate GP regression by exploiting the notion of a support set.

The *parallel partially independent training conditional* (pPITC) approximation of FGP model is adapted from our previous work on decentralized data fusion (Chen *et al.*, 2012) for sampling environmental phenomena with mobile sensors. But, the latter does not address the practical implementation issues of parallelization on a cluster of machines nor demonstrate scalability with large data. So, we present pPITC here under the setting of parallel machines and then show how its shortcomings can be overcome by extending it to pPIC. The key idea of pPITC is as follows: After distributing the data evenly among M machines (Step 1), each machine encapsulates its local data, based on a common prior support set $\mathcal{S} \subset \mathcal{X}$ where $|\mathcal{S}| \ll |\mathcal{D}|$, into a local summary that is communicated to the master[1] (Step 2). The master assimilates the local summaries into a global summary (Step 3), which is then sent back to the M machines to be used for predictions distributed among them (Step 4). These steps are detailed below:

STEP 1: DISTRIBUTE DATA AMONG M MACHINES.

The data $(\mathcal{D}, y_\mathcal{D})$ is partitioned evenly into M blocks, each of which is assigned to a machine, as defined below:

Definition 1 (Local Data) *The local data of machine m is defined as a tuple $(\mathcal{D}_m, y_{\mathcal{D}_m})$ where $\mathcal{D}_m \subseteq \mathcal{D}$, $\mathcal{D}_m \bigcap \mathcal{D}_i = \emptyset$ and $|\mathcal{D}_m| = |\mathcal{D}_i| = |\mathcal{D}|/M$ for $i \neq m$.*

STEP 2: EACH MACHINE CONSTRUCTS AND SENDS LOCAL SUMMARY TO MASTER.

Definition 2 (Local Summary) *Given a common support set $\mathcal{S} \subset \mathcal{X}$ known to all M machines and the local data $(\mathcal{D}_m, y_{\mathcal{D}_m})$, the local summary of machine m is defined as a tuple $(\dot{y}^m_\mathcal{S}, \dot{\Sigma}^m_{\mathcal{S}\mathcal{S}})$ where*

$$\dot{y}^m_\mathcal{B} \triangleq \Sigma_{\mathcal{B}\mathcal{D}_m}\Sigma^{-1}_{\mathcal{D}_m\mathcal{D}_m|\mathcal{S}}(y_{\mathcal{D}_m} - \mu_{\mathcal{D}_m}) \tag{3}$$

[1]One of the M machines can be assigned to be the master.

$$\dot{\Sigma}^m_{\mathcal{B}\mathcal{B}'} \triangleq \Sigma_{\mathcal{B}\mathcal{D}_m}\Sigma^{-1}_{\mathcal{D}_m\mathcal{D}_m|\mathcal{S}}\Sigma_{\mathcal{D}_m\mathcal{B}'} \tag{4}$$

such that $\Sigma_{\mathcal{D}_m\mathcal{D}_m|\mathcal{S}}$ is defined in a similar manner as (2) and $\mathcal{B}, \mathcal{B}' \subset \mathcal{X}$.

Remark. Since the local summary is independent of the outputs $y_\mathcal{S}$, they need not be observed. So, the support set $\mathcal{S}$ does not have to be a subset of $\mathcal{D}$ and can be selected prior to data collection. Predictive performances of pPITC and pPIC are sensitive to the selection of $\mathcal{S}$. An informative support set $\mathcal{S}$ can be selected from domain $\mathcal{X}$ using an iterative greedy active selection procedure (Krause *et al.*, 2008; Lawrence *et al.*, 2003; Seeger and Williams, 2003) prior to observing data. For example, the differential entropy score criterion (Lawrence *et al.*, 2003) can be used to greedily select an input $x \in \mathcal{X}\setminus\mathcal{S}$ with the largest posterior variance $\Sigma_{xx|\mathcal{S}}$ (2) to be included in $\mathcal{S}$ in each iteration.

STEP 3: MASTER CONSTRUCTS AND SENDS GLOBAL SUMMARY TO M MACHINES.

Definition 3 (Global Summary) *Given a common support set $\mathcal{S} \subset \mathcal{X}$ known to all M machines and the local summary $(\dot{y}^m_\mathcal{S}, \dot{\Sigma}^m_{\mathcal{S}\mathcal{S}})$ of every machine $m = 1, \ldots, M$, the global summary is defined as a tuple $(\ddot{y}_\mathcal{S}, \ddot{\Sigma}_{\mathcal{S}\mathcal{S}})$ where*

$$\ddot{y}_\mathcal{S} \triangleq \sum_{m=1}^{M} \dot{y}^m_\mathcal{S} \tag{5}$$

$$\ddot{\Sigma}_{\mathcal{S}\mathcal{S}} \triangleq \Sigma_{\mathcal{S}\mathcal{S}} + \sum_{m=1}^{M} \dot{\Sigma}^m_{\mathcal{S}\mathcal{S}} \,. \tag{6}$$

STEP 4: DISTRIBUTE PREDICTIONS AMONG M MACHINES.

To predict the unobserved outputs for any set $\mathcal{U}$ of inputs, $\mathcal{U}$ is partitioned evenly into disjoint subsets $\mathcal{U}_1, \ldots, \mathcal{U}_M$ to be assigned to the respective machines $1, \ldots, M$. So, $|\mathcal{U}_m| = |\mathcal{U}|/M$ for $m = 1, \ldots, M$.

Definition 4 (pPITC) *Given a common support set $\mathcal{S} \subset \mathcal{X}$ known to all M machines and the global summary $(\ddot{y}_\mathcal{S}, \ddot{\Sigma}_{\mathcal{S}\mathcal{S}})$, each machine m computes a predictive Gaussian distribution $\mathcal{N}(\widehat{\mu}_{\mathcal{U}_m}, \widehat{\Sigma}_{\mathcal{U}_m\mathcal{U}_m})$ of the unobserved outputs for the set $\mathcal{U}_m$ of inputs where*

$$\widehat{\mu}_{\mathcal{U}_m} \triangleq \mu_{\mathcal{U}_m} + \Sigma_{\mathcal{U}_m\mathcal{S}}\ddot{\Sigma}^{-1}_{\mathcal{S}\mathcal{S}}\ddot{y}_\mathcal{S} \tag{7}$$

$$\widehat{\Sigma}_{\mathcal{U}_m\mathcal{U}_m} \triangleq \Sigma_{\mathcal{U}_m\mathcal{U}_m} - \Sigma_{\mathcal{U}_m\mathcal{S}}\left(\Sigma^{-1}_{\mathcal{S}\mathcal{S}} - \ddot{\Sigma}^{-1}_{\mathcal{S}\mathcal{S}}\right)\Sigma_{\mathcal{S}\mathcal{U}_m} \,. \tag{8}$$

Theorem 1 [Chen *et al.* (2012)] *Let a common support set $\mathcal{S} \subset \mathcal{X}$ be known to all M machines. Let $\mathcal{N}(\mu^{\text{PITC}}_{\mathcal{U}|\mathcal{D}}, \Sigma^{\text{PITC}}_{\mathcal{U}\mathcal{U}|\mathcal{D}})$ be the predictive Gaussian distribution computed by the centralized partially independent training conditional (PITC) approximation of FGP model (Quiñonero-Candela and Rasmussen, 2005) where*

$$\mu^{\text{PITC}}_{\mathcal{U}|\mathcal{D}} \triangleq \mu_\mathcal{U} + \Gamma_{\mathcal{U}\mathcal{D}}\left(\Gamma_{\mathcal{D}\mathcal{D}} + \Lambda\right)^{-1}\left(y_\mathcal{D} - \mu_\mathcal{D}\right) \tag{9}$$

$$\Sigma^{\text{PITC}}_{\mathcal{U}\mathcal{U}|\mathcal{D}} \triangleq \Sigma_{\mathcal{U}\mathcal{U}} - \Gamma_{\mathcal{U}\mathcal{D}}\left(\Gamma_{\mathcal{D}\mathcal{D}} + \Lambda\right)^{-1}\Gamma_{\mathcal{D}\mathcal{U}} \tag{10}$$

such that

$$\Gamma_{\mathcal{B}\mathcal{B}'} \triangleq \Sigma_{\mathcal{B}\mathcal{S}}\Sigma^{-1}_{\mathcal{S}\mathcal{S}}\Sigma_{\mathcal{S}\mathcal{B}'} \tag{11}$$

and Λ is a block-diagonal matrix constructed from the M diagonal blocks of $\Sigma_{\mathcal{D}\mathcal{D}|\mathcal{S}}$, each of which is a matrix $\Sigma_{\mathcal{D}_m\mathcal{D}_m|\mathcal{S}}$ for $m = 1, \ldots, M$ where $\mathcal{D} = \bigcup_{m=1}^{M}\mathcal{D}_m$. Then, $\widehat{\mu}_\mathcal{U} = \mu^{\text{PITC}}_{\mathcal{U}|\mathcal{D}}$ and $\widehat{\Sigma}_{\mathcal{U}\mathcal{U}} = \Sigma^{\text{PITC}}_{\mathcal{U}\mathcal{U}|\mathcal{D}}$.

The proof of Theorem 1 is previously reported in (Chen *et al.*, 2012) and reproduced in Appendix A of Chen *et al.* (2013) to reflect our notations.

Remark. Since PITC generalizes the Bayesian Committee Machine (BCM) of Schwaighofer and Tresp (2002), pPITC generalizes parallel BCM (Ingram and Cornford, 2010), the latter of which assumes the support set $\mathcal{S}$ to be $\mathcal{U}$ (Quiñonero-Candela and Rasmussen, 2005). As a result, parallel BCM does not scale well with large $\mathcal{U}$.

Though pPITC scales very well with large data (Table 1), it can predict poorly due to (a) loss of information caused by summarizing the realized outputs and correlation structure of the original data; and (b) sparse coverage of $\mathcal{U}$ by the support set. We propose a novel parallel Gaussian process regression method called pPIC that combines the best of both worlds, that is, the predictive power of FGP and time efficiency of pPITC. pPIC is based on the following intuition: A machine can exploit its local data to improve the predictions of the unobserved outputs that are highly correlated with its data. At the same time, pPIC can preserve the time efficiency of pPITC by exploiting its idea of encapsulating information into local and global summaries.

Definition 5 (pPIC) *Given a common support set $\mathcal{S} \subset \mathcal{X}$ known to all M machines, the global summary $(\ddot{y}_\mathcal{S}, \ddot{\Sigma}_{\mathcal{S}\mathcal{S}})$, the local summary $(\dot{y}^m_\mathcal{S}, \dot{\Sigma}^m_{\mathcal{S}\mathcal{S}})$, and the local data $(\mathcal{D}_m, y_{\mathcal{D}_m})$, each machine m computes a predictive Gaussian distribution $\mathcal{N}(\widehat{\mu}^+_{\mathcal{U}_m}, \widehat{\Sigma}^+_{\mathcal{U}_m\mathcal{U}_m})$ of the unobserved outputs for the set $\mathcal{U}_m$ of inputs where*

$$\widehat{\mu}^+_{\mathcal{U}_m} \triangleq \mu_{\mathcal{U}_m} + \left(\Phi^m_{\mathcal{U}_m\mathcal{S}}\ddot{\Sigma}^{-1}_{\mathcal{S}\mathcal{S}}\ddot{y}_\mathcal{S} - \Sigma_{\mathcal{U}_m\mathcal{S}}\Sigma^{-1}_{\mathcal{S}\mathcal{S}}\dot{y}^m_\mathcal{S}\right) + \dot{y}^m_{\mathcal{U}_m} \tag{12}$$

$$\begin{aligned}\widehat{\Sigma}^+_{\mathcal{U}_m\mathcal{U}_m} \triangleq \Sigma_{\mathcal{U}_m\mathcal{U}_m} - \Big(&\Phi^m_{\mathcal{U}_m\mathcal{S}}\Sigma^{-1}_{\mathcal{S}\mathcal{S}}\Sigma_{\mathcal{S}\mathcal{U}_m} - \Sigma_{\mathcal{U}_m\mathcal{S}}\Sigma^{-1}_{\mathcal{S}\mathcal{S}}\dot{\Sigma}^m_{\mathcal{S}\mathcal{U}_m}\\ &- \Phi^m_{\mathcal{U}_m\mathcal{S}}\ddot{\Sigma}^{-1}_{\mathcal{S}\mathcal{S}}\Phi^m_{\mathcal{S}\mathcal{U}_m}\Big) - \dot{\Sigma}^m_{\mathcal{U}_m\mathcal{U}_m}\end{aligned} \tag{13}$$

such that

$$\Phi^m_{\mathcal{U}_m\mathcal{S}} \triangleq \Sigma_{\mathcal{U}_m\mathcal{S}} + \Sigma_{\mathcal{U}_m\mathcal{S}}\Sigma^{-1}_{\mathcal{S}\mathcal{S}}\dot{\Sigma}^m_{\mathcal{S}\mathcal{S}} - \dot{\Sigma}^m_{\mathcal{U}_m\mathcal{S}} \tag{14}$$

and $\Phi^m_{\mathcal{S}\mathcal{U}_m}$ is the transpose of $\Phi^m_{\mathcal{U}_m\mathcal{S}}$.

Remark 1. The predictive Gaussian mean $\widehat{\mu}^+_{\mathcal{U}_m}$ (12) and covariance $\widehat{\Sigma}^+_{\mathcal{U}_m\mathcal{U}_m}$ (13) of pPIC exploit both summary information (i.e., bracketed term) and local information (i.e., last term). In contrast, pPITC only exploits the global summary (see (7) and (8)).

Remark 2. To improve the predictive performance of pPIC, $\mathcal{D}$ and $\mathcal{U}$ should be partitioned into tuples of $(\mathcal{D}_1,\mathcal{U}_1),\ldots,(\mathcal{D}_M,\mathcal{U}_M)$ such that the outputs $y_{\mathcal{D}_m}$ and $Y_{\mathcal{U}_m}$ are as highly correlated as possible for $m=1,\ldots,M$. To achieve this, we employ a simple parallelized clustering scheme in our experiments: Each machine m randomly selects a cluster center from its local data $\mathcal{D}_m$ and informs the other machines about its chosen cluster center. Then, each input in $\mathcal{D}_m$ and $\mathcal{U}_m$ is simply assigned to the "nearest" cluster center i and sent to the corresponding machine i while being subject to the constraints of the new D_i and U_i not exceeding $|\mathcal{D}|/M$ and $|\mathcal{U}|/M$, respectively. More sophisticated clustering schemes can be utilized at the expense of greater time and communication complexity.

Remark 3. Predictive performances of pPITC and pPIC are improved by increasing size of $\mathcal{S}$ at the expense of greater time, space, and communication complexity (Table 1).

Theorem 2 *Let a common support set $\mathcal{S}\subset\mathcal{X}$ be known to all M machines. Let $\mathcal{N}(\mu^{\text{PIC}}_{\mathcal{U}|\mathcal{D}},\Sigma^{\text{PIC}}_{\mathcal{U}\mathcal{U}|\mathcal{D}})$ be the predictive Gaussian distribution computed by the centralized partially independent conditional (PIC) approximation of FGP model (Snelson, 2007) where*

$$\mu^{\text{PIC}}_{\mathcal{U}|\mathcal{D}} \triangleq \mu_{\mathcal{U}} + \widetilde{\Gamma}_{\mathcal{U}\mathcal{D}}\left(\Gamma_{\mathcal{D}\mathcal{D}}+\Lambda\right)^{-1}\left(y_{\mathcal{D}}-\mu_{\mathcal{D}}\right) \tag{15}$$

$$\Sigma^{\text{PIC}}_{\mathcal{U}\mathcal{U}|\mathcal{D}} \triangleq \Sigma_{\mathcal{U}\mathcal{U}} - \widetilde{\Gamma}_{\mathcal{U}\mathcal{D}}\left(\Gamma_{\mathcal{D}\mathcal{D}}+\Lambda\right)^{-1}\widetilde{\Gamma}_{\mathcal{D}\mathcal{U}} \tag{16}$$

and $\widetilde{\Gamma}_{\mathcal{D}\mathcal{U}}$ is the transpose of $\widetilde{\Gamma}_{\mathcal{U}\mathcal{D}}$ such that

$$\widetilde{\Gamma}_{\mathcal{U}\mathcal{D}} \triangleq \left(\widetilde{\Gamma}_{\mathcal{U}_i\mathcal{D}_m}\right)_{i,m=1,\ldots,M} \tag{17}$$

$$\widetilde{\Gamma}_{\mathcal{U}_i\mathcal{D}_m} \triangleq \begin{cases}\Sigma_{\mathcal{U}_i\mathcal{D}_m} & \text{if } i=m,\\ \Gamma_{\mathcal{U}_i\mathcal{D}_m} & \text{otherwise.}\end{cases} \tag{18}$$

Then, $\widehat{\mu}^{+}_{\mathcal{U}}=\mu^{\text{PIC}}_{\mathcal{U}|\mathcal{D}}$ and $\widehat{\Sigma}^{+}_{\mathcal{U}\mathcal{U}}=\Sigma^{\text{PIC}}_{\mathcal{U}\mathcal{U}|\mathcal{D}}$.

Its proof is given in Appendix B of Chen *et al.* (2013).

Remark 1. The equivalence results of Theorems 1 and 2 imply that the computational load of the centralized PITC and PIC approximations of FGP can be distributed among M parallel machines, hence improving the time efficiency and scalability of approximate GP regression (Table 1).

Remark 2. The equivalence results also shed some light on the underlying properties of pPITC and pPIC based on the structural assumptions of PITC and PIC, respectively: pPITC assumes that $Y_{\mathcal{D}_1},\ldots,Y_{\mathcal{D}_M},Y_{\mathcal{U}_1},\ldots,Y_{\mathcal{U}_M}$ are conditionally independent given $Y_{\mathcal{S}}$. In contrast, pPIC can predict the unobserved outputs $Y_{\mathcal{U}}$ better since it imposes a less restrictive assumption of conditional independence between $Y_{\mathcal{D}_1\bigcup\mathcal{U}_1},\ldots,Y_{\mathcal{D}_M\bigcup\mathcal{U}_M}$ given $Y_{\mathcal{S}}$. This assumption further supports an earlier remark just before Theorem 2 on clustering inputs $\mathcal{D}_m$ and $\mathcal{U}_m$ whose corresponding outputs are highly correlated for improving predictive performance of pPIC. Experimental results on two real-world datasets (Section 6) show that pPIC achieves predictive accuracy comparable to FGP and significantly better than pPITC, thus justifying the practicality of such an assumption.

4 Parallel Gaussian Process Regression using Incomplete Cholesky Factorization

In this section, we will present another parallel Gaussian process called pICF-based GP that distributes the computational load among parallel machines to achieve efficient and scalable approximate GP regression by exploiting incomplete Cholesky factorization (ICF). A fundamental step of pICF-based GP is to use ICF to approximate the covariance matrix $\Sigma_{\mathcal{D}\mathcal{D}}$ in (1) and (2) of FGP by a low-rank symmetric positive semidefinite matrix: $\Sigma_{\mathcal{D}\mathcal{D}}\approx F^{\top}F+\sigma_n^2 I$ where $F\in\mathbb{R}^{R\times|\mathcal{D}|}$ denotes the upper triangular incomplete Cholesky factor and $R\ll|\mathcal{D}|$ is the reduced rank. The steps of performing pICF-based GP are as follows:

STEP 1: DISTRIBUTE DATA AMONG M MACHINES.

This step is the same as that of pPITC and pPIC in Section 3.

STEP 2: RUN PARALLEL ICF TO PRODUCE INCOMPLETE CHOLESKY FACTOR AND DISTRIBUTE ITS STORAGE.

ICF can in fact be parallelized: Instead of using a column-based parallel ICF (Golub and Van Loan, 1996), our proposed pICF-based GP employs a row-based parallel ICF, the latter of which incurs lower time, space, and communication complexity. Interested readers are referred to (Chang *et al.*, 2007) for a detailed implementation of the row-based parallel ICF, which is beyond the scope of this paper. More importantly, it produces an upper triangular incomplete Cholesky factor $F\triangleq(F_1\cdots F_M)$ and each submatrix $F_m\in\mathbb{R}^{R\times|\mathcal{D}_m|}$ is stored distributedly on machine m for $m=1,\ldots,M$.

STEP 3: EACH MACHINE CONSTRUCTS AND SENDS LOCAL SUMMARY TO MASTER.

Definition 6 (Local Summary) *Given the local data $(\mathcal{D}_m,y_{\mathcal{D}_m})$ and incomplete Cholesky factor F_m, the local summary of machine m is defined as a tuple $(\dot{y}_m,\dot{\Sigma}_m,\Phi_m)$ where*

$$\dot{y}_m \triangleq F_m(y_{\mathcal{D}_m}-\mu_{\mathcal{D}_m}) \tag{19}$$

$$\dot{\Sigma}_m \triangleq F_m\Sigma_{\mathcal{D}_m\mathcal{U}} \tag{20}$$

$$\Phi_m \triangleq F_m F_m^{\top}\ . \tag{21}$$

STEP 4: MASTER CONSTRUCTS AND SENDS GLOBAL SUMMARY TO M MACHINES.

Definition 7 (Global Summary) *Given the local summary $(\dot{y}_m,\dot{\Sigma}_m,\Phi_m)$ of every machine $m=1,\ldots,M$, the* *global summary is defined as a tuple* $(\ddot{y}, \ddot{\Sigma})$ *where*

$$\ddot{y} \triangleq \Phi^{-1} \sum_{m=1}^{M} \dot{y}_m \tag{22}$$

$$\ddot{\Sigma} \triangleq \Phi^{-1} \sum_{m=1}^{M} \dot{\Sigma}_m \tag{23}$$

such that $\Phi \triangleq I + \sigma_n^{-2} \sum_{m=1}^{M} \Phi_m$.

Remark. If $|\mathcal{U}|$ is large, the computation of (23) can be parallelized by partitioning $\mathcal{U}$: Let $\dot{\Sigma}_m \triangleq (\dot{\Sigma}_m^1 \cdots \dot{\Sigma}_m^M)$ where $\dot{\Sigma}_m^i \triangleq F_m \Sigma_{\mathcal{D}_m \mathcal{U}_i}$ is defined in a similar way as (20) and $|\mathcal{U}|_i = |\mathcal{U}|/M$. So, in Step 3, instead of sending $\dot{\Sigma}_m$ to the master, each machine m sends $\dot{\Sigma}_m^i$ to machine i for $i = 1, \ldots, M$. Then, each machine i computes and sends $\ddot{\Sigma}_i \triangleq \Phi^{-1} \sum_{m=1}^{M} \dot{\Sigma}_m^i$ to every other machine to obtain $\ddot{\Sigma} = (\ddot{\Sigma}_1 \cdots \ddot{\Sigma}_M)$.

STEP 5: EACH MACHINE CONSTRUCTS AND SENDS PREDICTIVE COMPONENT TO MASTER.

Definition 8 (Predictive Component) *Given the local data* $(\mathcal{D}_m, y_{\mathcal{D}_m})$*, a component* $\dot{\Sigma}_m$ *of the local summary, and the global summary* $(\ddot{y}, \ddot{\Sigma})$*, the predictive component of machine* m *is defined as a tuple* $(\widetilde{\mu}_{\mathcal{U}}^m, \widetilde{\Sigma}_{\mathcal{U}\mathcal{U}}^m)$ *where*

$$\widetilde{\mu}_{\mathcal{U}}^m \triangleq \sigma_n^{-2} \Sigma_{\mathcal{U}\mathcal{D}_m} (y_{\mathcal{D}_m} - \mu_{\mathcal{D}_m}) - \sigma_n^{-4} \dot{\Sigma}_m^\top \ddot{y} \tag{24}$$

$$\widetilde{\Sigma}_{\mathcal{U}\mathcal{U}}^m \triangleq \sigma_n^{-2} \Sigma_{\mathcal{U}\mathcal{D}_m} \Sigma_{\mathcal{D}_m \mathcal{U}} - \sigma_n^{-4} \dot{\Sigma}_m^\top \ddot{\Sigma} \; . \tag{25}$$

STEP 6: MASTER PERFORMS PREDICTIONS.

Definition 9 (pICF-based GP) *Given the predictive component* $(\widetilde{\mu}_{\mathcal{U}}^m, \widetilde{\Sigma}_{\mathcal{U}\mathcal{U}}^m)$ *of every machine* $m = 1, \ldots, M$*, the master computes a predictive Gaussian distribution* $\mathcal{N}(\widetilde{\mu}_{\mathcal{U}}, \widetilde{\Sigma}_{\mathcal{U}\mathcal{U}})$ *of the unobserved outputs for any set* $\mathcal{U}$ *of inputs where*

$$\widetilde{\mu}_{\mathcal{U}} \triangleq \mu_{\mathcal{U}} + \sum_{m=1}^{M} \widetilde{\mu}_{\mathcal{U}}^m \tag{26}$$

$$\widetilde{\Sigma}_{\mathcal{U}\mathcal{U}} \triangleq \Sigma_{\mathcal{U}\mathcal{U}} - \sum_{m=1}^{M} \widetilde{\Sigma}_{\mathcal{U}\mathcal{U}}^m \; . \tag{27}$$

Remark. Predictive performance of pICF-based GP can be improved by increasing rank R at the expense of greater time, space, and communication complexity (Table 1).

Theorem 3 *Let* $\mathcal{N}(\mu_{\mathcal{U}|\mathcal{D}}^{\text{ICF}}, \Sigma_{\mathcal{U}\mathcal{U}|\mathcal{D}}^{\text{ICF}})$ *be the predictive Gaussian distribution computed by the centralized ICF approximation of FGP model where*

$$\mu_{\mathcal{U}|\mathcal{D}}^{\text{ICF}} \triangleq \mu_{\mathcal{U}} + \Sigma_{\mathcal{U}\mathcal{D}} (F^\top F + \sigma_n^2 I)^{-1} (y_{\mathcal{D}} - \mu_{\mathcal{D}}) \tag{28}$$

$$\Sigma_{\mathcal{U}\mathcal{U}|\mathcal{D}}^{\text{ICF}} \triangleq \Sigma_{\mathcal{U}\mathcal{U}} - \Sigma_{\mathcal{U}\mathcal{D}} (F^\top F + \sigma_n^2 I)^{-1} \Sigma_{\mathcal{D}\mathcal{U}} \; . \tag{29}$$

Then, $\widetilde{\mu}_{\mathcal{U}} = \mu_{\mathcal{U}|\mathcal{D}}^{\text{ICF}}$ *and* $\widetilde{\Sigma}_{\mathcal{U}\mathcal{U}} = \Sigma_{\mathcal{U}\mathcal{U}|\mathcal{D}}^{\text{ICF}}$.

Its proof is given in Appendix C of Chen *et al.* (2013).

Remark 1. The equivalence result of Theorem 3 implies that the computational load of the centralized ICF approximation of FGP can be distributed among the M parallel machines, hence improving the time efficiency and scalability of approximate GP regression (Table 1).

Remark 2. By approximating the covariance matrix $\Sigma_{\mathcal{D}\mathcal{D}}$ in (1) and (2) of FGP with $F^\top F + \sigma_n^2 I$, $\widetilde{\Sigma}_{\mathcal{U}\mathcal{U}} = \Sigma_{\mathcal{U}\mathcal{U}|\mathcal{D}}^{ICF}$ is not guaranteed to be positive semidefinite, hence rendering such a measure of predictive uncertainty not very useful. However, it is observed in our experiments (Section 6) that this problem can be alleviated by choosing a sufficiently large rank R.

5 Analytical Comparison

This section compares and contrasts the properties of the proposed parallel GPs analytically.

5.1 Time, Space, and Communication Complexity

Table 1 analytically compares the time, space, and communication complexity between pPITC, pPIC, pICF-based GP, PITC, PIC, ICF-based GP, and FGP based on the following assumptions: (a) These respective methods compute the predictive means (i.e., $\widehat{\mu}_{\mathcal{U}}$ (7), $\widehat{\mu}_{\mathcal{U}}^{+}$ (12), $\widetilde{\mu}_{\mathcal{U}}$ (26), $\mu_{\mathcal{U}|\mathcal{D}}^{\text{PITC}}$ (9), $\mu_{\mathcal{U}|\mathcal{D}}^{\text{PIC}}$ (15), $\mu_{\mathcal{U}|\mathcal{D}}^{\text{ICF}}$ (28), and $\mu_{\mathcal{U}|\mathcal{D}}$ (1)) and their corresponding predictive variances (i.e., $\widehat{\Sigma}_{xx}$ (8), $\widehat{\Sigma}_{xx}^{+}$ (13), $\widetilde{\Sigma}_{xx}$ (27), $\Sigma_{xx|\mathcal{D}}^{\text{PITC}}$ (10), $\Sigma_{xx|\mathcal{D}}^{\text{PIC}}$ (16), $\Sigma_{xx|\mathcal{D}}^{\text{ICF}}$ (29), and $\Sigma_{xx|\mathcal{D}}$ (2) for all $x \in \mathcal{U}$); (b) $|\mathcal{U}| < |\mathcal{D}|$ and recall $|\mathcal{S}|, R \ll |\mathcal{D}|$; (c) the data is already distributed among M parallel machines for pPITC, pPIC, and pICF-based GP; and (d) for MPI, a broadcast operation in the communication network of M machines incurs $\mathcal{O}(\log M)$ messages (Pjesivac-Grbovic *et al.*, 2007). The observations are as follows:

(a) Our pPITC, pPIC, and pICF-based GP improve the scalability of their centralized counterparts (respectively, PITC, PIC, and ICF-based GP) in the size $|\mathcal{D}|$ of data by distributing their computational loads among the M parallel machines.

(b) The speedups of pPITC, pPIC, and pICF-based GP over their centralized counterparts deviate further from ideal speedup with increasing number M of machines due to their additional $\mathcal{O}(|\mathcal{S}|^2 M)$ or $\mathcal{O}(R^2 M)$ time.

(c) The speedups of pPITC and pPIC grow with increasing size $|\mathcal{D}|$ of data because, unlike the additional $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{D}|)$ time of PITC and PIC that increase with more data, they do not have corresponding $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{D}|/M)$ terms.

(d) Our pPIC incurs additional $\mathcal{O}(|\mathcal{D}|)$ time and $\mathcal{O}((|\mathcal{D}|/M) \log M)$-sized messages over pPITC

Table 1: Comparison of time, space, and communication complexity between pPITC, pPIC, pICF-based GP, PITC, PIC, ICF-based GP, and FGP. Note that PITC, PIC, and ICF-based GP are, respectively, the centralized counterparts of pPITC, pPIC, and pICF-based GP, as proven in Theorems 1, 2, and 3.

GP	Time complexity	Space complexity	Communication complexity
pPITC	$\mathcal{O}\left(\|\mathcal{S}\|^2\left(\|\mathcal{S}\|+M+\frac{\|\mathcal{U}\|}{M}\right)+\left(\frac{\|\mathcal{D}\|}{M}\right)^3\right)$	$\mathcal{O}\left(\|\mathcal{S}\|^2+\left(\frac{\|\mathcal{D}\|}{M}\right)^2\right)$	$\mathcal{O}\left(\|\mathcal{S}\|^2\log M\right)$
pPIC	$\mathcal{O}\left(\|\mathcal{S}\|^2\left(\|\mathcal{S}\|+M+\frac{\|\mathcal{U}\|}{M}\right)+\left(\frac{\|\mathcal{D}\|}{M}\right)^3+\|\mathcal{D}\|\right)$	$\mathcal{O}\left(\|\mathcal{S}\|^2+\left(\frac{\|\mathcal{D}\|}{M}\right)^2\right)$	$\mathcal{O}\left(\left(\|\mathcal{S}\|^2+\frac{\|\mathcal{D}\|}{M}\right)\log M\right)$
pICF-based	$\mathcal{O}\left(R^2\left(R+M+\frac{\|\mathcal{D}\|}{M}\right)+R\|\mathcal{U}\|\left(M+\frac{\|\mathcal{D}\|}{M}\right)\right)$	$\mathcal{O}\left(R^2+R\frac{\|\mathcal{D}\|}{M}\right)$	$\mathcal{O}\left(\left(R^2+R\|\mathcal{U}\|\right)\log M\right)$
PITC	$\mathcal{O}\left(\|\mathcal{S}\|^2\|\mathcal{D}\|+\|\mathcal{D}\|\left(\frac{\|\mathcal{D}\|}{M}\right)^2\right)$	$\mathcal{O}\left(\|\mathcal{S}\|^2+\left(\frac{\|\mathcal{D}\|}{M}\right)^2\right)$	–
PIC	$\mathcal{O}\left(\|\mathcal{S}\|^2\|\mathcal{D}\|+\|\mathcal{D}\|\left(\frac{\|\mathcal{D}\|}{M}\right)^2+M\|\mathcal{D}\|\right)$	$\mathcal{O}\left(\|\mathcal{S}\|^2+\left(\frac{\|\mathcal{D}\|}{M}\right)^2\right)$	–
ICF-based	$\mathcal{O}\left(R^2\|\mathcal{D}\|+R\|\mathcal{U}\|\|\mathcal{D}\|\right)$	$\mathcal{O}(R\|\mathcal{D}\|)$	–
FGP	$\mathcal{O}\left(\|\mathcal{D}\|^3\right)$	$\mathcal{O}\left(\|\mathcal{D}\|^2\right)$	–

due to its parallelized clustering (see Remark 2 after Definition 5).

(e) Keeping the other variables fixed, an increasing number M of machines reduces the time and space complexity of pPITC and pPIC at a faster rate than pICF-based GP while increasing size $|\mathcal{D}|$ of data raises the time and space complexity of pICF-based GP at a slower rate than pPITC and pPIC.

(f) Our pICF-based GP distributes the memory requirement of ICF-based GP among the M parallel machines.

(g) The communication complexity of pICF-based GP depends on the number $|\mathcal{U}|$ of predictions whereas that of pPITC and pPIC are independent of it.

5.2 Online/Incremental Learning

Supposing new data $(\mathcal{D}', y_{\mathcal{D}'})$ becomes available, pPITC and pPIC do not have to run Steps 1 to 4 (Section 3) on the entire data $(\mathcal{D}\bigcup\mathcal{D}', y_{\mathcal{D}\bigcup\mathcal{D}'})$. The local and global summaries of the old data $(\mathcal{D}, y_{\mathcal{D}})$ can in fact be reused and assimilated with that of the new data, thus saving the need of recomputing the computationally expensive matrix inverses in (3) and (4) for the old data. The exact mathematical details are omitted due to lack of space. As a result, the time complexity of pPITC and pPIC can be greatly reduced in situations where new data is expected to stream in at regular intervals. In contrast, pICF-based GP does not seem to share this advantage.

5.3 Structural Assumptions

The above advantage of online learning for pPITC and pPIC results from their assumptions of conditional independence (see Remark 2 after Theorem 2) given the support set. With fewer machines, such an assumption is violated less, thus potentially improving their predictive performances. In contrast, the predictive performance of pICF-based GP is not affected by varying the number of machines. However, it suffers from a different problem: Utilizing a reduced-rank matrix approximation of $\Sigma_{\mathcal{DD}}$, its resulting predictive covariance matrix $\widetilde{\Sigma}_{\mathcal{UU}}$ is not guaranteed to be positive semidefinite (see Remark 2 after Theorem 3), thus rendering such a measure of predictive uncertainty not very useful. It is observed in our experiments (Section 6) that this problem can be alleviated by choosing a sufficiently large rank R.

6 Experiments and Discussion

This section empirically evaluates the predictive performances, time efficiency, scalability, and speedups of our proposed parallel GPs against their centralized counterparts and FGP on two real-world datasets: (a) The AIMPEAK dataset of size $|\mathcal{D}| = 41850$ contains traffic speeds (km/h) along 775 road segments of an urban road network (including highways, arterials, slip roads, etc.) during the morning peak hours (6-10:30 a.m.) on April 20, 2011. The traffic speeds are the outputs. The mean speed is 49.5 km/h and the standard deviation is 21.7 km/h. Each input (i.e., road segment) is specified by a 5-dimensional vector of features: length, number of lanes, speed limit, direction, and time. The time dimension comprises 54 five-minute time slots. This spatiotemporal traffic phenomenon is modeled using a relational GP (previously developed in (Chen *et al.*, 2012)) whose correlation structure can exploit both the road segment features and road network topology information; (b) The SARCOS dataset (Vijayakumar *et al.*, 2005) of size $|\mathcal{D}| = 48933$ pertains to an inverse dynamics problem for a seven degrees-of-freedom SARCOS robot arm. Each input denotes a 21-dimensional vector of features: 7 joint positions, 7 joint velocities, and 7 joint accelerations. Only one of the 7 joint torques is used as the output. The mean torque is 13.7 and the standard deviation is 20.5.

Both datasets are modeled using GPs whose prior covariance $\sigma_{xx'}$ is defined by the squared exponential covariance function[2]:

$$\sigma_{xx'} \triangleq \sigma_s^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{d}\left(\frac{x_i - x'_i}{\ell_i}\right)^2\right) + \sigma_n^2 \delta_{xx'}$$

where $x_i\,(x'_i)$ is the i-th component of the input feature vector $x\,(x')$, the hyperparameters $\sigma_s^2, \sigma_n^2, \ell_1, \ldots, \ell_d$ are, respectively, signal variance, noise variance, and length-scales; and $\delta_{xx'}$ is a Kronecker delta that is 1 if $x = x'$ and 0 otherwise. The hyperparameters are learned using randomly selected data of size 10000 via maximum likelihood estimation (Rasmussen and Williams, 2006).

For each dataset, 10% of the data is randomly selected as test data for predictions (i.e., as $\mathcal{U}$). From the remaining data, training data of varying sizes $|\mathcal{D}| =$ 8000, 16000, 24000, and 32000 are randomly selected. The training data are distributed among M machines based on the simple parallelized clustering scheme in Remark 2 after Definition 5. Our pPITC and pPIC are evaluated using support sets of varying sizes $|\mathcal{S}| =$ 256, 512, 1024, and 2048 that are selected using differential entropy score criterion (see remark just after Definition 2). Our pICF-based GP is evaluated using varying reduced ranks R of the same values as $|\mathcal{S}|$ in the AIMPEAK domain and twice the values of $|\mathcal{S}|$ in the SARCOS domain.

Our experimental platform is a cluster of 20 computing nodes connected via gigabit links: Each node runs a Linux system with Intel® Xeon® CPU E5520 at 2.27 GHz and 20 GB memory. Our parallel GPs are tested with different number $M =$ 4, 8, 12, 16, and 20 of computing nodes.

6.1 Performance Metrics

The tested GP regression methods are evaluated with four different performance metrics: (a) Root mean square error (RMSE) $\sqrt{|\mathcal{U}|^{-1}\sum_{x\in\mathcal{U}}\left(y_x - \mu_{x|\mathcal{D}}\right)^2}$; (b) mean negative log probability (MNLP) $0.5|\mathcal{U}|^{-1}\sum_{x\in\mathcal{U}}\left((y_x - \mu_{x|\mathcal{D}})^2/\Sigma_{xx|\mathcal{D}} + \log(2\pi\Sigma_{xx|\mathcal{D}})\right)$ (Rasmussen and Williams, 2006); (c) incurred time; and (d) speedup is defined as the incurred time of a sequential/centralized algorithm divided by that of its corresponding parallel algorithm. For the first two metrics, the tested methods have to plug their predictive mean and variance into $\mu_{u|\mathcal{D}}$ and $\Sigma_{uu|\mathcal{D}}$, respectively.

[2]For the AIMPEAK dataset, the domain of road segments is embedded into the Euclidean space using multi-dimensional scaling (Chen *et al.*, 2012) so that a squared exponential covariance function can then be applied.

6.2 Results and Analysis

In this section, we analyze the results that are obtained by averaging over 5 random instances.

6.2.1 Varying size $|\mathcal{D}|$ of data

Figs. 1a-b and 1e-f show that the predictive performances of our parallel GPs improve with more data and are comparable to that of FGP, hence justifying the practicality of their inherent structural assumptions.

From Figs. 1e-f, it can be observed that the predictive performance of pICF-based GP is very close to that of FGP when $|\mathcal{D}|$ is relatively small (i.e., $|\mathcal{D}| =$ 8000, 16000). But, its performance approaches that of pPIC as $|\mathcal{D}|$ increases further because the reduced rank $R = 4096$ of pICF-based GP is not large enough (relative to $|\mathcal{D}|$) to maintain its close performance to FGP. In addition, pPIC achieves better predictive performance than pPITC since the former can exploit local information (see Remark 1 after Definition 5).

Figs. 1c and 1g indicate that our parallel GPs are significantly more time-efficient and scalable than FGP (i.e., 1-2 orders of magnitude faster) while achieving comparable predictive performance. Among the three parallel GPs, pPITC and pPIC are more time-efficient and thus more capable of meeting the real-time prediction requirement of a time-critical application/system.

Figs. 1d and 1h show that the speedups of our parallel GPs over their centralized counterparts increase with more data, which agree with observation c in Section 5.1. pPITC and pPIC achieve better speedups than pICF-based GP.

6.2.2 Varying number M of machines

Figs. 2a-b and 2e-f show that pPIC and pICF-based GP achieve predictive performance comparable to that of FGP with different number M of machines. pPIC achieves better predictive performance than pPITC due to its use of local information (see Remark 1 after Definition 5).

From Figs. 2e-f, it can be observed that as the number M of machines increases, the predictive performance of pPIC drops slightly due to smaller size of local data $\mathcal{D}_m$ assigned to each machine. In contrast, the predictive performance of pPITC improves: If the number M of machines is small as compared to the actual number of clusters in the data, then the clustering scheme (see Remark 2 after Definition 5) may assign data from different clusters to the same machine or data from the same cluster to multiple machines. Consequently, the conditional independence assumption is violated. Such an issue is mitigated by increasing the number M of machines to achieve better clustering, hence resulting in better predictive performance.

Figs. 2c and 2g show that pPIC and pICF-based GP are significantly more time-efficient than FGP (i.e., 1-2 orders

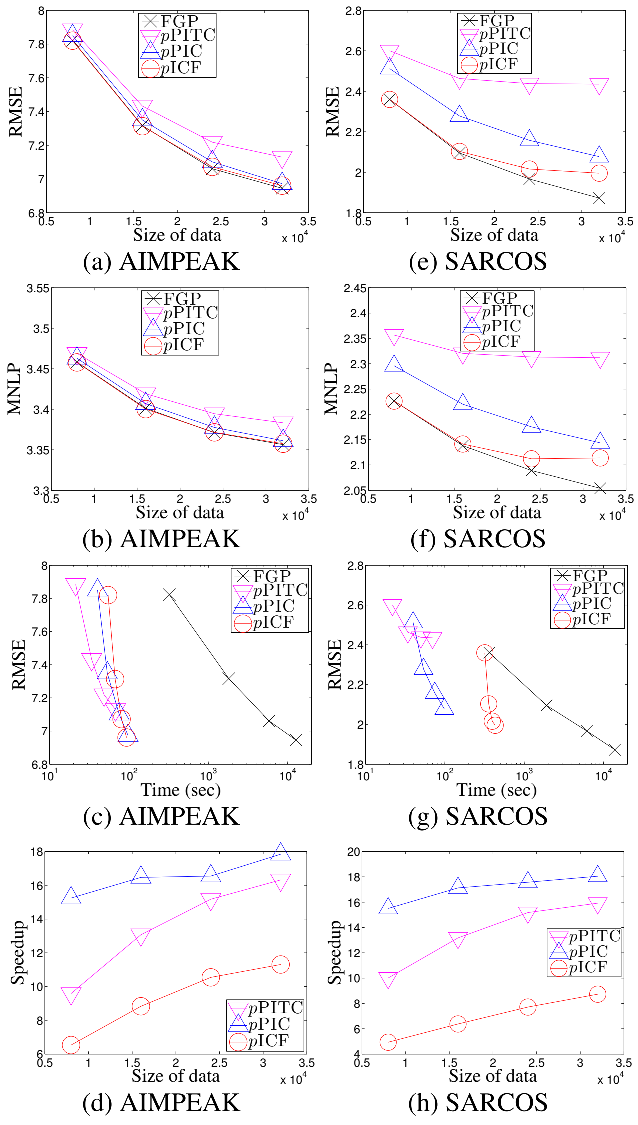


Figure 1: Performance of parallel GPs with varying data sizes $|\mathcal{D}| = 8000$, 16000, 24000, and 32000, number $M = 20$ of machines, support set size $|\mathcal{S}| = 2048$, and reduced rank $R = 2048$ (4096) in the AIMPEAK (SARCOS) domain.

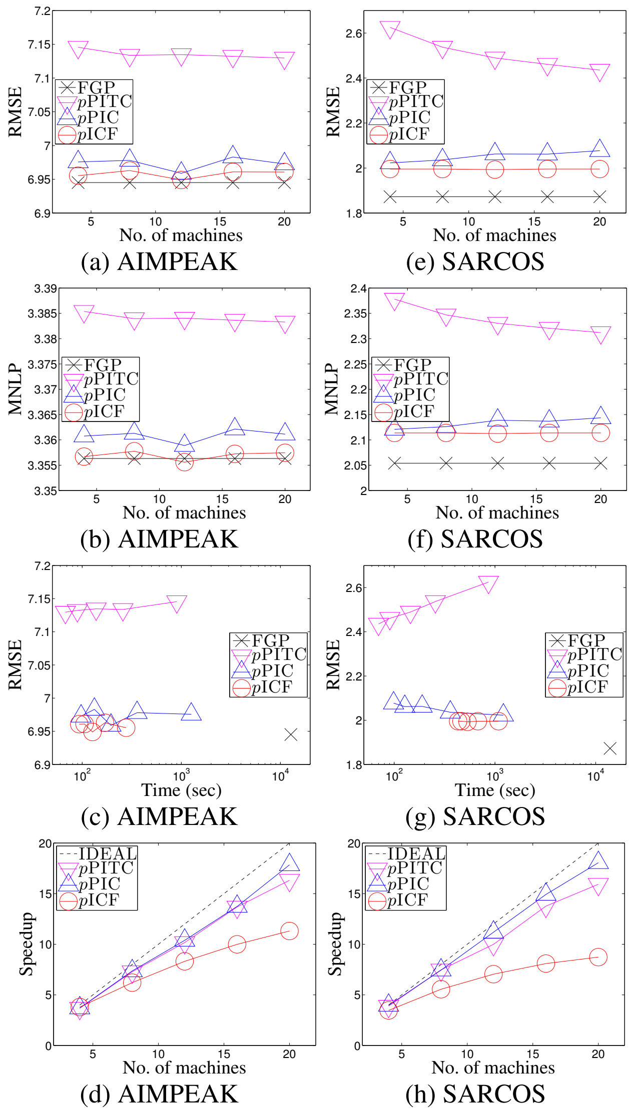


Figure 2: Performance of parallel GPs with varying number $M = 4, 8, 12, 16, 20$ of machines, data size $|\mathcal{D}| = 32000$, support set size $\mathcal{S} = 2048$, and reduced rank $R = 2048$ (4096) in the AIMPEAK (SARCOS) domain. The *ideal* speedup of a parallel algorithm is defined to be the number M of machines running it.

of magnitude faster) while achieving comparable predictive performance. This is previously explained in the analysis of their time complexity (Table 1).

Figs. 2c and 2g also reveal that as the number M of machines increases, the incurred time of pPITC and pPIC decreases at a faster rate than that of pICF-based GP, which agree with observation e in Section 5.1. Hence, we expect pPITC and pPIC to be more time-efficient than pICF-based GP when the number M of machines increases beyond 20.

Figs. 2d and 2h show that the speedups of our parallel GPs over their centralized counterparts deviate further from the ideal speedup with a greater number M of machines, which agree with observation b in Section 5.1. The speedups of pPITC and pPIC are closer to the ideal speedup than that of pICF-based GP.

6.2.3 Varying support set size $|\mathcal{S}|$ and reduced rank R

Figs. 3a and 3e show that the predictive performance of pICF-based GP is extremely poor when the reduced rank R is not large enough (relative to $|\mathcal{D}|$), thus resulting in a poor ICF approximation of the covariance matrix $\Sigma_{\mathcal{D}\mathcal{D}}$. In addition, it can be observed that the reduced rank R of pICF-based GP needs to be much larger than the support set size $|\mathcal{S}|$ of pPITC and pPIC in order to achieve comparable predictive performance. These results also indicate that the heuristic $R = \sqrt{|\mathcal{D}|}$, which is used by Chang *et al.* (2007) to determine the reduced rank R, fails to work well in both our datasets (e.g., $R = 1024 > \sqrt{32000} \approx 179$).

From Figs. 3b and 3f, it can be observed that pICF-based GP incurs negative MNLP for $R \le 1024$ ($R \le 2048$) in the AIMPEAK (SARCOS) domain. This is because pICF-based GP cannot guarantee positivity of predictive variance, as explained in Remark 2 after Theorem 3. But, it appears that when R is sufficiently large (i.e., $R = 2048$ ($R = 4096$) in the AIMPEAK (SARCOS) domain), this problem can be alleviated.

It can be observed in Figs. 3c and 3g that pPITC and pPIC are significantly more time-efficient than FGP (i.e., 2-4 orders of magnitude faster) while achieving comparable predictive performance. To ensure high predictive performance, pICF-based GP has to select a large enough rank $R = 2048$ ($R = 4096$) in the AIMPEAK (SARCOS) domain, thus making it less time-efficient than pPITC and pPIC. But, it can still incur 1-2 orders of magnitude less time than FGP. These results indicate that pPITC and pPIC are more capable than pICF-based GP of meeting the real-time prediction requirement of a time-critical application/system.

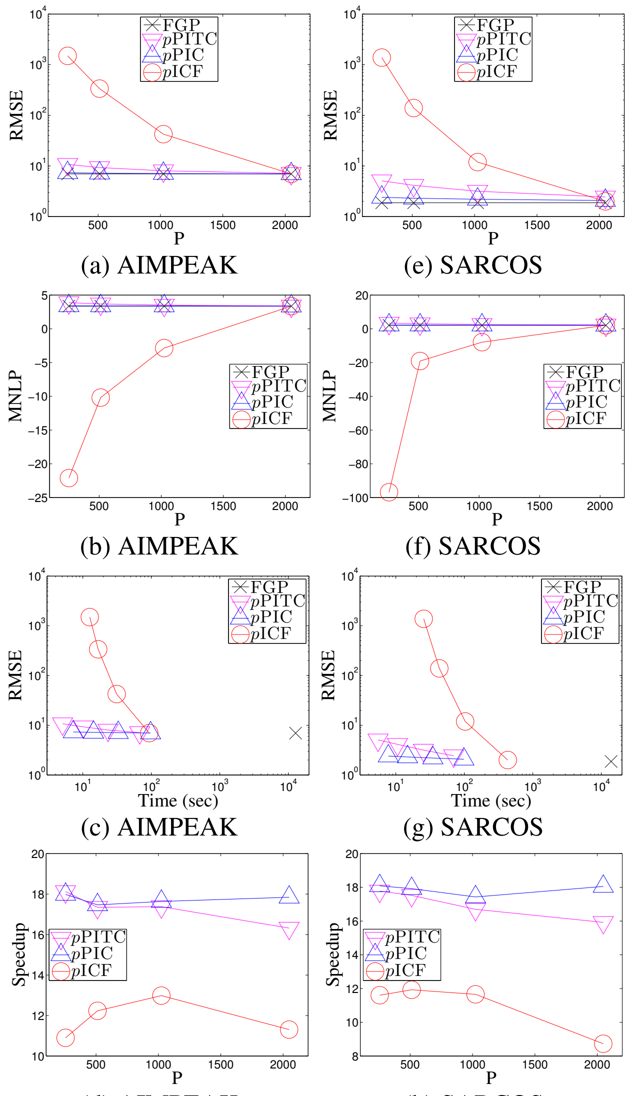
Figs. 3d and 3h show that pPITC and pPIC achieve better speedups than pICF-based GP.

6.2.4 Summary of results

pPIC and pICF-based GP are significantly more time-efficient and scalable than FGP (i.e., 1-4 orders of magnitude faster) while achieving comparable predictive performance, hence justifying the practicality of their structural assumptions. pPITC and pPIC are expected to be more time-efficient than pICF-based GP with an increasing number M of machines because their incurred time decreases at a faster rate than that of pICF-based GP. Since the predictive performances of pPITC and pPIC drop slightly (i.e., more stable) with smaller $|\mathcal{S}|$ as compared to that of pICF-based GP dropping rapidly with smaller R, pPITC and pPIC are more capable than pICF-based GP of meeting the real-time prediction requirement of time-critical applications. The speedups of our parallel GPs over their centralized counterparts improve with more data but deviate further from ideal speedup with larger number of machines.

7 Conclusion

This paper describes parallel GP regression methods called pPIC and pICF-based GP that, respectively, distribute the computational load of the centralized PIC and ICF-based GP among parallel machines to achieve greater time efficiency and scalability. Analytical and empirical results have demonstrated that our parallel GPs are significantly more time-efficient and scalable than their centralized counterparts and FGP while achieving predictive performance comparable to FGP. As a result, by exploiting large clusters of machines, our parallel GPs become substantially more capable of performing real-time predictions necessary in many time-critical applications/systems. We have also implemented pPITC and pPIC in the MapReduce framework for running in a Linux server with 2 Intel® Xeon® CPU E5-2670 at 2.60 GHz and 96 GB memory (i.e., 16 cores); due to shared memory, they incur slightly longer time than that in a cluster of 16 computing nodes. We plan to release the source code at http://code.google.com/p/pgpr/.



(a) AIMPEAK (e) SARCOS
(b) AIMPEAK (f) SARCOS
(c) AIMPEAK (g) SARCOS
(d) AIMPEAK (h) SARCOS

Figure 3: Performance of parallel GPs with data size $|\mathcal{D}| = 32000$, number $M = 20$ of machines, and varying parameter $P = 256, 512, 1024, 2048$ where $P = |\mathcal{S}| = R$ ($P = |\mathcal{S}| = R/2$) in the AIMPEAK (SARCOS) domain.

Acknowledgments. This work was supported by Singapore-MIT Alliance Research & Technology Subaward Agreements No. 28 R-252-000-502-592 & No. 33 R-252-000-509-592.

References

Bekkerman, R., Bilenko, M., and Langford, J. (2011). *Scaling up Machine Learning: Parallel and Distributed Approaches*. Cambridge Univ. Press, NY.

Cao, N., Low, K. H., and Dolan, J. M. (2013). Multi-robot informative path planning for active sensing of environmental phenomena: A tale of two algorithms. In *Proc. AAMAS*, pages 7–14.

Chang, E. Y., Zhu, K., Wang, H., Bai, H., Li, J., Qiu, Z., and Cui, H. (2007). Parallelizing support vector machines on distributed computers. In *Proc. NIPS*.

Chen, J., Low, K. H., Tan, C. K.-Y., Oran, A., Jaillet, P., Dolan, J. M., and Sukhatme, G. S. (2012). Decentralized data fusion and active sensing with mobile sensors for modeling and predicting spatiotemporal traffic phenomena. In *Proc. UAI*, pages 163–173.

Chen, J., Cao, N., Low, K. H., Ouyang, R., Tan, C. K.-Y., and Jaillet, P. (2013). Parallel Gaussian process regression with low-rank covariance matrix approximations. arXiv:1305.5826.

Choudhury, A., Nair, P. B., and Keane, A. J. (2002). A data parallel approach for large-scale Gaussian process modeling. In *Proc. SDM*, pages 95–111.

Das, K. and Srivastava, A. N. (2010). Block-GP: Scalable Gaussian process regression for multimodal data. In *Proc. ICDM*, pages 791–796.

Dolan, J. M., Podnar, G., Stancliff, S., Low, K. H., Elfes, A., Higinbotham, J., Hosler, J. C., Moisan, T. A., and Moisan, J. (2009). Cooperative aquatic sensing using the telesupervised adaptive ocean sensor fleet. In *Proc. SPIE Conference on Remote Sensing of the Ocean, Sea Ice, and Large Water Regions*, volume 7473.

Furrer, R., Genton, M. G., and Nychka, D. (2006). Covariance tapering for interpolation of large spatial datasets. *JCGS*, **15**(3), 502–523.

Golub, G. H. and Van Loan, C.-F. (1996). *Matrix Computations*. Johns Hopkins Univ. Press, third edition.

Ingram, B. and Cornford, D. (2010). Parallel geostatistics for sparse and dense datasets. In P. M. Atkinson and C. D. Lloyd, editors, *Proc. geoENV VII*, pages 371–381. Quantitative Geology and Geostatistics Volume 16, Springer, Netherlands.

Krause, A., Singh, A., and Guestrin, C. (2008). Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *JMLR*, **9**, 235–284.

Lawrence, N. D., Seeger, M., and Herbrich, R. (2003). Fast sparse Gaussian process methods: The informative vector machine. In *Advances in Neural Information Processing Systems 15*, pages 609–616. MIT Press.

Low, K. H., Gordon, G. J., Dolan, J. M., and Khosla, P. (2007). Adaptive sampling for multi-robot wide-area exploration. In *Proc. IEEE ICRA*, pages 755–760.

Low, K. H., Dolan, J. M., and Khosla, P. (2011). Active Markov information-theoretic path planning for robotic environmental sensing. In *Proc. AAMAS*, pages 753–760.

Low, K. H., Chen, J., Dolan, J. M., Chien, S., and Thompson, D. R. (2012). Decentralized active robotic exploration and mapping for probabilistic field classification in environmental sensing. In *Proc. AAMAS*, pages 105–112.

Park, C., Huang, J. Z., and Ding, Y. (2011). Domain decomposition approach for fast Gaussian process regression of large spatial data sets. *JMLR*, **12**, 1697–1728.

Pjesivac-Grbovic, J., Angskun, T., Bosilca, G., Fagg, G. E., Gabriel, E., and Dongarra, J. (2007). Performance analysis of MPI collective operations. *Cluster Computing*, **10**(2), 127–143.

Podnar, G., Dolan, J. M., Low, K. H., and Elfes, A. (2010). Telesupervised remote surface water quality sensing. In *Proc. IEEE Aerospace Conference*.

Quiñonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *JMLR*, **6**, 1939–1959.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA.

Schwaighofer, A. and Tresp, V. (2002). Transductive and inductive methods for approximate Gaussian process regression. In *Proc. NIPS*, pages 953–960.

Seeger, M. and Williams, C. (2003). Fast forward selection to speed up sparse Gaussian process regression. In *Proc. AISTATS*.

Snelson, E. (2007). Local and global sparse Gaussian process approximations. In *Proc. AISTATS*.

Snelson, E. and Ghahramani, Z. (2005). Sparse Gaussian processes using pseudo-inputs. In *Proc. NIPS*.

Vanhatalo, J. and Vehtari, A. (2008). Modeling local and global phenomena with sparse Gaussian processes. In *Proc. UAI*, pages 571–578.

Vijayakumar, S., D'Souza, A., and Schaal, S. (2005). Incremental online learning in high dimensions. *Neural Comput.*, **17**(12), 2602–2634.

Williams, C. K. I. and Seeger, M. (2000). Using the Nyström method to speed up kernel machines. In *Proc. NIPS*, pages 682–688.

Yu, J., Low, K. H., Oran, A., and Jaillet, P. (2012). Hierarchical Bayesian nonparametric approach to modeling and learning the wisdom of crowds of urban traffic route planning agents. In *Proc. IAT*, pages 478–485.

Convex Relaxations of Bregman Divergence Clustering

Hao Cheng
Department of Computing Science
University of Alberta
hcheng2@ualberta.ca

Xinhua Zhang
Machine Learning Research Group
National ICT Australia and ANU
xinhua.zhang@nicta.com.au

Dale Schuurmans
Department of Computing Science
University of Alberta
dale@cs.ualberta.ca

Abstract

Although many convex relaxations of clustering have been proposed in the past decade, current formulations remain restricted to spherical Gaussian or discriminative models and are susceptible to imbalanced clusters. To address these shortcomings, we propose a new class of convex relaxations that can be flexibly applied to more general forms of Bregman divergence clustering. By basing these new formulations on *normalized* equivalence relations we retain additional control on relaxation quality, which allows improvement in clustering quality. We furthermore develop optimization methods that improve scalability by exploiting recent implicit matrix norm methods. In practice, we find that the new formulations are able to efficiently produce tighter clusterings that improve the accuracy of state of the art methods.

1 Introduction

Discovering latent class structure in data, *i.e. clustering*, is a fundamental problem in machine learning and statistics. Given data, the task is to assign each observation a latent cluster label or distribution over cluster labels. Clustering has a long history, with diverse approaches proposed. Unfortunately, computational tractability remains a fundamental challenge: standard clustering formulations are *NP*-hard (Aloise et al., 2009; Dasgupta, 2008; Arora & Kannan, 2005) and additional problem structure must be postulated before efficient solutions can be guaranteed. Fortunately, standard clustering formulations are also efficiently approximable (Kumar et al., 2004), and much work has sought practical algorithms that improve solution quality, even in lieu of theoretical bounds. In this paper we contribute a new family of convex relaxations that improve clustering quality while admitting efficient algorithms.

The techniques we propose are applicable to a variety of clustering formulations. Two of the most important paradigms for clustering are based on *generative* versus *discriminative* modeling, with generative clustering consisting of hard clustering with conditional models, hard clustering with joint models, and soft clustering with joint models. We address all but soft clustering in this paper.

Traditionally, clustering formulations have used *generative models* to discover interesting latent structure in data. Let $\mathbf{X}$ denote the observation variable and $\mathbf{Y}$ denote the latent class variable. The simplest generative approach optimizes the conditional model $P(\mathbf{X}|\mathbf{Y})$ only, with $\mathbf{Y}$ assigned to the most likely value. This is also known as *hard conditional* clustering. When $P(\mathbf{X}|\mathbf{Y})$ is Gaussian, a popular approach is hard k-means (MacQueen, 1967) where one alternates between optimizing $\mathbf{Y}$ and the model. Banerjee et al. (2005) extended the formulation to general exponential family forms for $P(\mathbf{X}|\mathbf{Y})$ via Bregman divergences. Although hard conditional clustering provides a standard baseline, finding global solutions in this case is intractable; efficient methods are only known when the number of clusters or the dimensionality of the space is constrained (Hansen et al., 1998; Inaba et al., 1994). Consequently, there has been significant work on developing approximations, particularly via convex relaxations that can be solved in polynomial time. For example, Zha et al. (2001) derived a convex quadratic reformulation of conditional Gaussian clustering, and Peng & Wei (2007) obtained a tighter semidefinite programming (SDP) relaxation. By analyzing the complete positivity (CP) properties of the resulting constraint, Zass & Shashua (2005) propose an approximation for Gaussian clustering based on CP factorization. These can be further extended to relaxations of normalized graph-cut clustering (Xing & Jordan, 2003; Ng et al., 2001). Unfortunately, all these relaxations are restricted to Gaussian $P(\mathbf{X}|\mathbf{Y})$, and the optimization algorithms depend heavily on the linearity of the SDP objective.

The conditional clustering approach can be extended to *hard joint* clustering by explicitly including the class prior, thus optimizing the joint likelihood $P(\mathbf{X}, \mathbf{Y})$ with the most likely $\mathbf{Y}$. Again, efficient solution methods are not generally known, leaving local approaches as the only option.

To smooth these objectives, the *soft joint* model optimizes the marginal likelihood, $P(\mathbf{X}) = \sum_Y P(\mathbf{Y})P(\mathbf{X}|\mathbf{Y})$ (Neal & Hinton, 1998; Banerjee et al., 2005), which has traditionally been tackled by expectation-maximization (EM) (Dempster et al., 1977). The EM algorithm remains susceptible to local optima however. Intensive research has been devoted to understanding properties of the Gaussian mixture model in particular (Moitra & Valiant, 2010; Kalai et al., 2010; Dasgupta & Schulman, 2007; Chaudhuri et al., 2009). Although run time can be reduced to polynomial when the number of clusters or data dimensionality is constrained, it remains exponential in these quantities jointly. A few convex relaxations for soft joint clustering models have therefore been proposed. For example, Lashkari & Golland (2007) restrict cluster centers to data points, while Nowozin & Bakir (2008) exert sparsity inducing regularization over the class priors (while still embedding an intractable subproblem). Recent spectral techniques can provably recover an approximate estimate of Gaussian mixtures in polynomial time (Hsu & Kakade, 2013; Anandkumar et al., 2012). Unfortunately, this formulation remains restricted to spherical Gaussian forms of $P(\mathbf{X}|\mathbf{Y})$.

Finally, *discriminative models* provide a distinct paradigm for clustering that can be more effective when the goal of learning is to predict labels from the observation $\mathbf{X}$, *e.g.* as in semi-supervised learning (Chapelle et al., 2006). In this approach, one maximizes the reverse conditional likelihood $P(\mathbf{Y}|\mathbf{X})$, with $\mathbf{Y}$ imputed by the most likely label. A straightforward optimization strategy can alternate between optimizing $\mathbf{Y}$ and the model, but this quickly leads to local optima. Thus, here too, convex relaxation has been a popular approximation strategy, either in the case of a large margin loss (Xu & Schuurmans, 2005) or logistic loss (Joulin & Bach, 2012; Joulin et al., 2010; Bach & Harchaoui, 2007; Guo & Schuurmans, 2007). To date, such formulations have been entirely based on SDP relaxations with *unnormalized* equivalence matrices, whose elements indicate whether two examples belong to the same cluster. Such an approach is hampered by imbalanced clustering, since the model employs no mechanism to avoid assigning all examples to a single cluster.

In this paper we present new convex relaxations for hard conditional, hard joint, and discriminative clustering. One of the key results is a tighter convex relaxation of hard generative models for Bregman divergence clustering that also accounts for cluster size. We design efficient new algorithms that optimize the resulting *nonlinear* SDPs using recent induced matrix norm techniques. By applying standard rounding methods, we observe that the resulting clustering algorithms deliver lower sum of intra-cluster divergences and more faithful alignment with class labels in practice. Finally, applying our formulation to discriminative models immediately leads to normalized equivalence relations, which automatically alleviate the problem of imbalanced cluster assignment faced by current relaxations.

2 Background

Following (Banerjee et al., 2005), we formulate clustering as maximum likelihood estimation in an exponential family model with a latent variable $\mathbf{Y} \in \{1, \ldots, d\}$ (the class indicator). The observed variable $\mathbf{X}$ is in $\mathbb{R}^n$, from which an *iid* sample $X = (\mathbf{x}_1, \ldots, \mathbf{x}_t)'$ has been collected.

Generative models. In generative modeling we parameterize the joint distribution over $(\mathbf{X}, \mathbf{Y})$ as $\mathbf{Y} \to \mathbf{X}$:

$$p(\mathbf{Y} = j) = q_j, \tag{1}$$

$$p(\mathbf{X} = \mathbf{x}|\mathbf{Y} = j) = \exp\left(-D_F(\mathbf{x}, \boldsymbol{\mu}_j)\right) Z_j(\mathbf{x}). \tag{2}$$

Here $\Theta := \{q_j, \boldsymbol{\mu}_j\}_{j=1}^d$ are the parameters, where $\mathbf{q} \in \Delta_d$, the d dimensional simplex. We assume $P(\mathbf{X}|\mathbf{Y})$ is an exponential family model defined by the Bregman divergence D_F, where F is a strictly convex function with gradient $f = \nabla F$ (the transfer function), such that

$$D_F(\mathbf{x}, \mathbf{y}) := F(\mathbf{x}) - F(\mathbf{y}) - \langle \mathbf{x} - \mathbf{y}, f(\mathbf{y}) \rangle \,. \tag{3}$$

Here it is known that $D_F(\mathbf{x}, \mathbf{y}) = D_{F^*}(f(\mathbf{y}), f(\mathbf{x}))$, where F^* is the Fenchel conjugate of F. Also, f^{-1} is well defined by the strict convexity of F, and $f^{-1} = \nabla F^*$. Examples of commonly used Bregman divergences include Euclidean ($f(x) = x$), and sigmoid ($f(x) = \log \frac{x}{1-x}$).

Given data X, the parameters Θ can be estimated via

$$\operatorname*{argmax}_{\Theta} \max_{Y} p(X, Y|\Theta) \tag{4}$$

$$\text{or} \quad \operatorname*{argmax}_{\Theta}\; p(X|\Theta) = \max_{\Theta} \sum_{Y} p(X, Y|\Theta), \tag{5}$$

depending on whether Y is to be maximized (hard clustering) or summed out (soft clustering). Here we are letting Y denote a $t \times d$ assignment matrix such that $Y_{ij} \in \{0, 1\}$ and $Y\mathbf{1} = \mathbf{1}$ (a vector of all 1's with proper dimension). If we additionally let $\Gamma = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_d)$ and $B = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$, such that $\mathbf{b}_j = f(\boldsymbol{\mu}_j)$, then the conditional likelihood (2) can be rewritten over the entire data set as

$$p(X|Y) = \exp\left(-D_F(X, Y\Gamma)\right) Z(X) \tag{6}$$

$$= \exp\left(-D_{F^*}(YB, f(X))\right) Z(X), \tag{7}$$

where $D_F(X, Y\Gamma) := \sum_{i=1}^t D_F(X_{i:}, Y_{i:}\Gamma)$ and $D_{F^*}(YB, f(X)) := \sum_{i=1}^t D_{F^*}(Y_{i:}B, f(X_{i:}))$ are row-wise sums such that $X_{i:}$ stands for the i-th row of X.

Discriminative models. As an alternative, discriminative clustering uses a graphical model $\mathbf{X} \to \mathbf{Y}$, and focuses on modeling the dependence of the labels Y given X:

$$p(Y|X; W, \mathbf{b}) = \exp(-D_{F^*}(Y, f(XW + \mathbf{1b}')))Z(X),$$

where $\mathbf{b} \in \mathbb{R}^d$ is the offset for all clusters. A soft clustering model cannot be applied in this case, since $\sum_Y p(X, Y) = p(X)$. Instead, hard optimization of Y leads to

$$\min_{W, \mathbf{b}, Y} D_F(XW + \mathbf{1b}', f^{-1}(Y)). \tag{8}$$

All of these problems involve a mix of discrete and continuous variables, which raises considerable challenges. Our goal is to develop convex relaxations that can be solved efficiently while leading (after rounding) to higher quality solutions than those obtained by naive local optimization.

3 Conditional Generative Clustering

We first consider the case of *hard conditional clustering*, where the prior $\mathbf{q}$ has been fixed to some value beforehand.

3.1 Case 1: Jointly Convex Bregman Divergence

First note that by using (6), the estimator (4) can be reduced to $\min_{Y,\Gamma} D_F(X, Y\Gamma)$. Here Banerjee et al. (2005) showed that for any fixed assignment Y the optimal Γ is given by $\Gamma = (Y'Y)^\dagger Y'X$, for any Bregman divergence D_F. Plugging the solution back into the formulation, the problem becomes $\min_Y D_F(X, Y(Y'Y)^\dagger Y'X)$. Let us introduce the *normalized equivalence matrix*

$$M = Y(Y'Y)^\dagger Y' = Y \operatorname{diag}(Y'\mathbf{1})^\dagger Y', \tag{9}$$

where $\mathcal{M}$ is the set of possibilities. It then suffices to solve

$$\min_{M\in\mathcal{M}} D_F(X, MX). \tag{10}$$

This problem remains challenging for two reasons. First, the objective is not convex in M, since D_F is only guaranteed to be convex in its first argument. However, many Bregman divergences are *jointly* convex in both arguments; *e.g.* Mahalonobis distance, KL divergence, Bernoulli entropy, Bose-Einstein entropy, Itakura-Saito distortion, and von Neumann divergence (Wang & Schuurmans, 2003; Tsuda et al., 2004). We consider this simpler case first.

The second challenge lies in the non-convexity of the constraint set $\mathcal{M}$. Peng & Wei (2007) have shown that

$$\mathcal{M} = \left\{M : M = M', M^2 = M, \operatorname{tr}(M) \le d, M_{i:} \in \Delta_t\right\}.$$

Since $M^2 = M$ is the source of non-convexity, its convex hull can be used to construct a convex outer approximation of $\mathcal{M}$ (note that this is *not* taking the convex hull of $\mathcal{M}$):

$$\begin{aligned}\mathcal{M}_1 &:= \operatorname{conv}\left\{M : M = M' = M^2\right\} \cap \left\{M \in \Delta_t^t : \operatorname{tr}(M) \le d\right\}\\ &= \left\{M : \mathbf{0} \preceq M \preceq I, \operatorname{tr}(M) \le d, M_{i:} \in \Delta_t\right\},\end{aligned}$$

where by $M \succeq \mathbf{0}$ we also encode $M = M'$. Note that $M \preceq I$ is implied by $\mathbf{0} \preceq M$ and $M_{i:} \in \Delta_t$ (*e.g.* Mirsky, 1955, Theorem 7.5.4). Conveniently, $\mathcal{M}_1$ can be relaxed further by keeping only the spectral constraints

$$\mathcal{M}_2 := \left\{M : \mathbf{0} \preceq M \preceq I, \operatorname{tr}(M) \le d, M\mathbf{1} = \mathbf{1}\right\}.$$

Although this set $\mathcal{M}_1$ has been widely used, it is still not clear whether it is the tightest convex relaxation of $\mathcal{M}$; that is, whether $\mathcal{M}_1 = \operatorname{conv}\mathcal{M}$? With some surprise, we show that this conjecture is not true in Appendix A.

3.1.1 Optimization

Assuming D_F is convex in its second argument, one can easily minimize $D_F(X, MX)$ over $M \in \mathcal{M}_1$ by using the alternating direction method of multipliers (ADMM) (Boyd et al., 2010). In particular, we split the constraints into two groups: spectral and non-spectral, leading to the following augmented Lagrangian:

$$\begin{aligned}\mathcal{L}(M, Z, \Lambda) = D_F(X, MX) + \delta(M_{i:} \in \Delta_t) + \delta(Z \in \mathcal{M}_2)\\ - \langle \Lambda, M - Z\rangle + \frac{1}{2\mu}\|M - Z\|_F^2,\end{aligned}$$

where $\delta(\cdot) = 0$ if $\cdot$ is true; ∞ otherwise. The ADMM then proceeds as follows in each iteration:

1. $M_t \leftarrow \operatorname{argmin}_M \mathcal{L}(M, Z_{t-1}, \Lambda_{t-1})$; i.e. optimize objective under non-spectral constraints.

2. $Z_t \leftarrow \operatorname{argmin}_Z \mathcal{L}(M_t, Z, \Lambda_{t-1})$; i.e. project to satisfy the spectral constraints.

3. $\Lambda_t \leftarrow \Lambda_{t-1} + \frac{1}{\mu}(Z_t - M_t)$; i.e. update the multipliers.

Note that since we constrain $M_{i:} \in \Delta_t$, the objective $D_F(X, MX)$ remains well defined in Step 1. Furthermore, since the objective decomposes row-wise, each row of M can be optimized independently, which constitutes a key advantage of this scheme. Second, since Step 2 merely involves projection onto spectral constraints $\mathcal{M}_2$, a closed form solution exists based on eigen-decomposition, as established in the following lemma.

Lemma 1. *Let* $H = I - \frac{1}{t}\mathbf{1}\mathbf{1}'$. *Then*

$$\mathcal{M}_2 = \left\{HMH + \tfrac{1}{t}\mathbf{1}\mathbf{1}' : M \in \mathcal{M}_3\right\}, \tag{11}$$

where $\mathcal{M}_3 = \{M : \mathbf{0} \preceq M \preceq I, \operatorname{tr}(M) \le d - 1\}$. (12)

Proof. Clearly the right-hand side of (11) is contained in $\mathcal{M}_2$. Conversely, for any $M_2 \in \mathcal{M}_2$, we construct an $M \in \mathcal{M}_3$ as $M = M_2 - \frac{1}{t}\mathbf{1}\mathbf{1}'$. Note that $M_2\mathbf{1} = \mathbf{1}$ implies $\mathbf{1}/\sqrt{t}$ is an eigenvector of M_2 with eigenvalue 1. Therefore $M \succeq \mathbf{0}$. The rest is easy to verify. ☐

By Proposition 1, the problem of projecting any matrix A to $\mathcal{M}_2$ can be accomplished by solving

$$\min_{Z\in\mathcal{M}_2} \|Z - A\|^2 = \min_{S\in\mathcal{M}_3} \left\|HSH - (A - \tfrac{1}{t}\mathbf{1}\mathbf{1}')\right\|^2.$$

Let $B = A - \frac{1}{t}\mathbf{1}\mathbf{1}'$ and $V = B - HBH$. Then $HVH = \mathbf{0}$, hence the probem reduces to solving

$$\min_{S\in\mathcal{M}_3} \|HSH - HBH - V\|^2 = \min_{S\in\mathcal{M}_3} \|HSH - HBH\|^2 + \|V\|^2.$$

Now it suffices to solve $\min_{T\in\mathcal{M}_3} \|T - HBH\|^2$ *and* show the optimal T satisfies $HTH = T$. Suppose HBH has eigenvalues σ_i and eigenvectors ϕ_i. Then the optimal T must have eigenvalues μ_i and eigenvectors ϕ_i such that

$$\min_{\mu_i} \sum_i (\mu_i - \sigma_i)^2, \text{ s.t. } \mu_i \in [0,1], \sum_i \mu_i \le d - 1. \tag{13}$$

Since $\mathbf{1}$ is an eigenvector of HBH with eigenvalue 0, it is trivial that the corresponding μ_i in the optimal solution is also 0. Therefore, $T\mathbf{1} = \mathbf{0}$ and $HTH = T$. Finally the optimal Z is simply given by $T + \frac{1}{t}\mathbf{1}\mathbf{1}'$.

3.2 Case 2: Arbitrary Bregman Divergence

When the Bregman divergence is not convex in its second argument, we require a more general treatment. The key idea we exploit is to introduce a regularizer that allows a useful form of representer theorem to be applied. In particular, we augment the negative log likelihood of $P(Y|X)$ in (7) with a regularizer on the basis B, weighted by the number of points in the corresponding cluster. The resulting objective can be written:

$$\min_{Y,B} D_{F^*}(YB, f(X)) + \frac{\alpha}{2}\|YB\|_F^2. \tag{14}$$

Note B must be in the range of f. By the representer theorem, there exists a matrix $A \in \mathbb{R}^{t\times n}$ such that the optimal B can be written $B = (Y'Y)^\dagger Y'A$, which yields

$$\min_{M,A} D_{F^*}(MA, f(X)) + \frac{\alpha}{2}\operatorname{tr}(A'MA), \tag{15}$$

where M is defined in (9). We will work with this formulation by relaxing the domain of M to $\mathcal{M}_2$. Extension to $M \in \mathcal{M}_1$ is also straightforward by ADMM.

3.2.1 Optimization

Although (15) does not immediately exhibit joint convexity in M and A, a change of variable immediately leads to a convex formulation. Denote $T = MA$, then $\operatorname{Im}(T) \subseteq \operatorname{Im}(M)$ where $\operatorname{Im}(M)$ is the range of M. Also, denote $L(Z) := D_{F^*}(Z, f(X))$ for clarity.

Proposition 2. *The problem* (15) *is equivalent to*

$$\min_{M\in\mathcal{M}_3}\ \min_{T:\operatorname{Im}(T)\subseteq\operatorname{Im}(M)} L(T) + \frac{\alpha}{2}\operatorname{tr}(T'M^\dagger T) \tag{16}$$

$$= \min_T L(T) + \frac{\alpha}{2}\underbrace{\min_{M\in\mathcal{M}_3:\operatorname{Im}(T)\subseteq\operatorname{Im}(M)} \operatorname{tr}(T'M^\dagger T)}_{:=\Omega^2(T),\text{ with }\Omega(T)\geq 0}. \tag{17}$$

That is, any optimal (M, A) for (15) *provides an optimal solution to* (16) *via $T = MA$. Conversely, given any optimal (M,T) for* (16)*, $\operatorname{Im}(T) \subseteq \operatorname{Im}(M)$ guarantees $T = MA$ for some A. Thus (M, A) is optimal for* (15).

This proposition allows one to solve a convex problem in T, provided that $\Omega^2(T)$ is convex and easy to compute. Interestingly, $\Omega(T)$ has other favorable properties to exploit.

Theorem 3. *$\Omega(T)$ defines a norm on T. Ω and its dual norm Ω_* can be computed in $O(t^3)$ and $O(t^2 d)$ time resp.*[1]

With these conclusions, we can optimize (17) using a generalized conditional gradient method, accelerated by local search (Laue, 2012; Zhang et al., 2012); see Algorithm 1 (further details are given in Appendix C). At each iteration, the algorithm employs a linear approximation of L The inner oracle searches for a steepest descent direction by computing a subgradient of the dual norm Ω_*. Algorithm 1 is guaranteed to find an ϵ accurate solution to (17) in $O(1/\epsilon)$ iterations; see e.g. (Zhang et al., 2012). The optimal M can then be recovered by evaluating Ω at the optimal T.[2]

Algorithm 1 Conditional gradient for optimizing (17)

```
1: Initialize $T_0 = \mathbf{0}$. $s_0 = 0$.
2: for $k = 0, 1, \ldots$ do
3:     Set $S_k \in \partial\Omega_*(\nabla L(T_k))$, i.e. find a minimizer of
       $\min_S \langle\nabla L(T_k), S\rangle + \frac{\alpha}{2}\Omega^2(S)$ up to scaling.
4:     Line search:
       $(a, b) := \operatorname{argmin}_{a\geq 0, b\geq 0} L(aT_k + bS_k) + \frac{\alpha}{2}(as_k + b)^2$.
5:     Set $T_{k+1} = aT_k + bS_k$, $s_{k+1} = as_k + b$.
6: end for
```

We prove Theorem 3 in three steps.

1. Computing Ω. Let the singular values of T be $s_1 \geq \ldots \geq s_t$. Since $\Omega^2(T) = \min_{M\in\mathcal{M}_3}\operatorname{tr}(TT'M^\dagger)$, by von Neumann's trace inequality (Mirsky, 1975) the optimal M must have eigenvectors equal to the left singular vectors of T. The minimal objective value is then $\sum_i s_i^2/\sigma_i$, where σ_i are the eigenvalues of M. It suffices to solve

$$f(\mathbf{s}) := \min_{\{\sigma_i\}} \sum_{i=1}^t \frac{s_i^2}{\sigma_i},\ s.t.\ \sigma_i \in [0,1],\ \sum_{i=1}^t \sigma_i \leq d-1 \tag{18}$$

$$= \min_{\sigma_i\in[0,1]}\max_{\lambda\geq 0}\sum_{i=1}^t \frac{s_i^2}{\sigma_i} + \lambda\left(1 - d + \sum_{i=1}^t \sigma_i\right) \tag{19}$$

$$= \max_{\lambda\geq 0}\left\{\lambda(1-d) + \min_{\sigma_i\in[0,1]}\sum_{i=1}^t\left(\frac{s_i^2}{\sigma_i} + \lambda\sigma_i\right)\right\}. \tag{20}$$

Fixing λ, the optimal σ_i is attained at $\sigma_i(\lambda) = \frac{s_i}{\sqrt{\lambda}}$ if $\lambda \geq s_i^2$, and 1 if $\lambda < s_i^2$. Note that $\sigma_i(\lambda)$ decreases monotonically for $\lambda \geq s_t^2$, hence we only need to find a λ that satisfies $\sum_{i=1}^t \sigma_i(\lambda) = d - 1$, since the constraint $\sum_i \sigma_i \leq d - 1$ must be equality at the optimum. This only requires a line search over λ, which can be conducted efficiently as follows. Suppose the optimal λ lies in $[s_k^2, s_{k+1}^2]$. Then $\sigma_i(\lambda) = 1$ for all $i \leq k$ and $\sigma_i(\lambda) = s_i/\sqrt{\lambda}$ for all $i > k$. So $k + \frac{1}{\sqrt{\lambda}}\sum_{i=k+1}^t s_i = d - 1$, hence

$$\sqrt{\lambda} = \frac{1}{d-1-k}\sum_{i=k+1}^t s_i \in [s_k, s_{k+1}] \Rightarrow \begin{cases} k + \frac{\sum_{i=k+1}^t s_i}{s_k} \leq d-1 \\ k + \frac{\sum_{i=k+1}^t s_i}{s_{k+1}} \geq d-1. \end{cases}$$

Now note there must be a k satisfying these two conditions. Since both $k + \frac{1}{s_k}\sum_{i=k+1}^t s_i$ and $k + \frac{1}{s_{k+1}}\sum_{i=k+1}^t s_i$ grow monotonically in k, the smallest k that satisfies the second condition must also satisfy the first condition. Hence the optimal solution is $\sigma_i = 1$ for all $i \leq k$, and $\sigma_i = (d - 1 - k)s_i/\sum_{i=k+1}^t s_i$ for $i > k$.

[1] The same conclusion holds for $M \in \mathcal{M}_2$ (see Appendix B).

[2] This solution is valid since (16) minimizes over M and T. If the problem were $\min_T \max_M$ instead, the optimal M could not be generally recovered by maximizing M for fixed optimal T.

Algorithm 2 Compute $f(\mathbf{s})$ with given d.

```
1: for $k = 0, 1, \dots, d-2$ do
2:     if $\sum_{i=k+1}^{t} s_i \geq (d-1-k)s_{k+1}$ then break
3: end for
4: Return $f(\mathbf{s}) = \sum_{i=1}^{k} s_i^2 + \frac{1}{d-1-k}\left(\sum_{i=k+1}^{t} s_i\right)^2$.
```

The algorithm for evaluating $f(\mathbf{s}) = \Omega^2(T)$ is given in Algorithm 2. The 'if' condition in step 2 must be met when $k = d-2$. The computational cost is dominated by a full SVD of T, and fortunately our method needs to compute $\Omega(T)$ only once at the optimal T.

2. Ω is a norm. Note that $\Omega(T)$ depends only on the singular values of T. So it suffices to show that $\kappa(\mathbf{s}) := \sqrt{f(\mathbf{s})}$ is a symmetric gauge (Horn & Johnson, 1985, Theorem 3.5.18), where $f(\mathbf{s})$ is defined in (18). Clearly $\kappa(\mathbf{s})$ is permutation invariant, $\kappa(a\mathbf{s}) = |a|\,\kappa(\mathbf{s})$ for all $a \in \mathbb{R}$, and $\kappa(\mathbf{s}) = 0$ iff $\mathbf{s} = \mathbf{0}$. So it suffices to prove the triangle inequality for $\kappa(\mathbf{s})$. For any $\mathbf{s}_1$ and $\mathbf{s}_2$, let $t_1 = \kappa(\mathbf{s}_1)$ and $t_2 = \kappa(\mathbf{s}_2)$. Then $\kappa(\frac{\mathbf{s}_1}{t_1}) = \kappa(\frac{\mathbf{s}_2}{t_2}) = 1$, and

$$\frac{\mathbf{s}_1 + \mathbf{s}_2}{t_1 + t_2} = \frac{t_1}{t_1 + t_2}\frac{\mathbf{s}_1}{t_1} + \frac{t_2}{t_1 + t_2}\frac{\mathbf{s}_2}{t_2}. \qquad (21)$$

Note $f(\mathbf{s})$ is convex because $\sum_i s_i^2/\sigma_i$ is jointly convex in $(\mathbf{s}, \boldsymbol{\sigma})$, and $f(\mathbf{s})$ just minimizes out $\boldsymbol{\sigma}$. So the sub-level set at level 1 for f (and κ) is convex. Therefore by (21), $\kappa((\mathbf{s}_1 + \mathbf{s}_2)/(t_1 + t_2)) \leq 1$, and so $\kappa(\mathbf{s}_1 + \mathbf{s}_2) \leq t_1 + t_2 = \kappa(\mathbf{s}_1) + \kappa(\mathbf{s}_2)$. The claim follows.

3. Compute the subgradient of Ω_*. Given a matrix R, the dual norm is $\Omega_*(R) = \max_{T:\Omega(T)\leq 1} \mathrm{tr}(R'T)$. Let the SVD of R be $R = U\,\mathrm{diag}\{r_1, \dots, r_t\}V'$, where $r_1 \geq \dots \geq r_t$. Since Ω is defined via the singular values of T, again by von Neumann's trace inequality the maximum is attained when the left and right singular values of T are U and V, respectively. Then $\Omega_*(R) = \max_{\mathbf{s}: f(\mathbf{s})\leq 1} \mathbf{r}'\mathbf{s}$, which by (18) is equivalent to

$$\max_{\mathbf{s},\boldsymbol{\sigma}} \mathbf{r}'\mathbf{s}, \ \ s.t.\ \sigma_i \in [0,1],\ \sum_{i=1}^{t}\sigma_i \leq d-1,\ \sum_{i=1}^{t}\frac{s_i^2}{\sigma_i} \leq 1. \ (22)$$

Using the Cauchy-Schwarz inequality, we have

$$\mathbf{r}'\mathbf{s} = \sum_{i=1}^{t}\frac{s_i}{\sqrt{\sigma_i}}\cdot r_i\sqrt{\sigma_i} \leq \left(\sum_{i=1}^{t}\frac{s_i^2}{\sigma_i}\right)^{1/2}\left(\sum_{i=1}^{t} r_i^2\sigma_i\right)^{1/2}$$
$$\leq \left(\sum_{i=1}^{t} r_i^2\sigma_i\right)^{1/2} \leq \|(r_1, r_2, \dots, r_{d-1})'\|. \qquad (23)$$

where the last two inequalities use the constraints in (22). The equalities can all be attained by setting $s_i = r_i/\|(r_1, r_2, \dots, r_{d-1})'\|$ and $\sigma_i = 1$ for $i \leq d-1$, and $s_i = 0$ and $\sigma_i = 0$ for $i \geq d$. Clearly $U\,\mathrm{diag}(\mathbf{s})V'$ is a subgradient of Ω_* at R. Evaluating the dual norm is inexpensive, since it requires only the top $d-1$ singular values of R.

4 Discriminative Clustering

Although generative models can often reveal useful latent structure in data, many problems such as semi-supervised learning and multiple instance learning are more concerned with accurate label prediction. In such settings, discriminative models $\mathbf{X} \to \mathbf{Y}$ can often be more effective (Joulin & Bach, 2012; Bach & Harchaoui, 2007; Guo & Schuurmans, 2007; Xu & Schuurmans, 2005).

Before attempting a convex relaxation for the discriminative model (8), it is important to recognize that a plain optimization over $(W, \mathbf{b}, Y)$ will lead to vacuous solutions, where all examples are assigned to a single cluster j and $b_j = \infty$. A common solution is to add a regularizer on Y to enforce a more balanced cluster distribution. Note that this situation is opposite of generative clustering, where one must upper bound d, since otherwise the joint likelihood would be trivially maximized by assigning each data point to its own cluster.

For discriminative clustering, we consider a special case $F(\mathbf{x}) = \log\sum_i \exp(x_i)$, *i.e.* where the transfer ∇F is sigmoidal (Joulin & Bach, 2012). A natural choice of regularizer on Y is the entropy of cluster sizes, *i.e.* $-h(Y'\mathbf{1})$ where $h(\mathbf{x}) = \sum_i x_i \log x_i$. In this setting, we derive a convex relaxation for discriminative clustering that uses the normalized equivalence matrix.

By adding value regularization $\|WY'\|^2$ to (8), one obtains

$$\min_{W,\mathbf{b},Y} \frac{1}{t} D_F(XW + \mathbf{1b}', f^{-1}(Y)) + \frac{\gamma}{2}\|WY'\|^2 + h(Y'\mathbf{1})$$
$$= \min_{W,\mathbf{b},Y} \frac{1}{t}F(XW + \mathbf{1b}') - \frac{1}{t}\mathrm{tr}((XW + \mathbf{1b}')Y')$$
$$- \frac{1}{t}F(Y) + \frac{\gamma}{2}\|WY'\|^2 + h(Y'\mathbf{1})$$
$$= \min_{W,\mathbf{b},Y} \max_{\Lambda:\Lambda_{i:}\in\Delta} -\frac{1}{t}F^*(\Lambda) + \frac{1}{t}\mathrm{tr}(\Lambda'(XW + \mathbf{1b}'))$$
$$- \frac{1}{t}F(Y) - \mathrm{tr}((XW + \mathbf{1b}')Y') + \frac{\gamma}{2}\|WY'\|^2 + h(Y'\mathbf{1})$$
$$= \min_{W,\mathbf{b},Y} \max_{\Omega:\Omega_{i:}\in\Delta} -\frac{1}{t}F^*(\Omega Y) + \frac{1}{t}\mathrm{tr}(Y'\Omega'(XW + \mathbf{1b}'))$$
$$- \frac{1}{t}F(Y) - \frac{1}{t}\mathrm{tr}((XW + \mathbf{1b}')Y') + \frac{\gamma}{2}\|WY'\|^2 + h(Y'\mathbf{1}).$$

Here, the second step follows from Fenchel's identity $F(\mathbf{x}) = \max_{\mathbf{z}\in\mathrm{dom}\,F^*} \mathbf{x}'\mathbf{z} - F^*(\mathbf{z})$, where dom denotes the effective domain of a convex function. The last step involves a change of variable, $\Lambda = \Omega Y$, and converted the constraints on Λ to $\Omega_{i:} \in \Delta$ (Guo & Schuurmans, 2007). By taking the gradient with respect to W and $\mathbf{b}$, one obtains

$$W = \tfrac{1}{t}X'(I - \Omega)Y(Y'Y)^\dagger, \text{ and } \Omega'\mathbf{1} = \mathbf{1}. \qquad (24)$$

Note that $-\frac{1}{t}F^*(\Omega Y) + h(Y'\mathbf{1}) \leq -\frac{1}{t}F^*(\Omega) + c_0$ where c_0 is some constant (Joulin & Bach, 2012, Eq 3). Using

(24) and the fact that $F(Y)$ is a constant, one can upper bound the objective by

$$\min_{M\in\mathcal{M}} \max_{\Omega:\Omega_{i:}\in\Delta,\Omega'\mathbf{1}=\mathbf{1}} -\frac{1}{t}F^*(\Omega)-\frac{1}{2\gamma t^2}\|X'(I-\Omega)M\|^2. \quad (25)$$

Importantly, this formulation is expressed completely in terms of the normalized equivalence matrix M, which constitutes a significant advantage over (Joulin & Bach, 2012; Guo & Schuurmans, 2007). Rather than resort to the proximal gradient method to solve for Ω given M (Joulin & Bach, 2012), which is slow in practice, we can harness the power of second order solvers like L-BFGS by dualizing the problem back to the primal form, which leads to an unconstrained problem. This reformulation also sheds light on the nature of the relaxation (25).

Fixing $M \in \mathcal{M}$, we add a Lagrange multiplier $\boldsymbol{\tau} \in \mathbb{R}^t$ to enforce $\Omega'\mathbf{1} = \mathbf{1}$. By introducing the change of variable $\Psi = I - \Omega$, the optimization over Ω becomes equivalent to

$$\min_{\Psi\le I:\Psi\mathbf{1}=\mathbf{0}} \frac{1}{t}F^*(I-\Psi)+\frac{1}{2\gamma t^2}\|X'\Psi M\|^2+\frac{1}{t}\boldsymbol{\tau}'\Psi\mathbf{1}. \quad (26)$$

The tool we use for dualization is provided by the following lemma.

Lemma 4. (Borwein & Lewis, 2000, Theorem 3.3.5) *Let J and G be convex functions, and A a linear transform. Suppose $A\operatorname{dom} J$ has nonempty intersection with $\{\mathbf{x}\in\operatorname{dom}G^* : G^*$ is continuous at $\mathbf{x}\}$. Then*

$$\min_{\mathbf{x}} J(\mathbf{x})+G(A\mathbf{x}) = \max_{\mathbf{y}} -J^*(-A'\mathbf{y})-G^*(\mathbf{y}). \quad (27)$$

To apply Lemma 4 to (26), choose the linear transform A to be $\Psi \mapsto \frac{1}{t}X'\Psi M$, $G(\Psi) = \frac{1}{2\gamma}\operatorname{tr}(\Psi M^\dagger\Psi')$,[3] and $J(\Psi) = \frac{1}{t}F^*(I-\Psi)+\frac{1}{t}\boldsymbol{\tau}'\Psi\mathbf{1}$ over $\Psi\mathbf{1}=\mathbf{0}$ and $\Psi \le I$ (elementwise). Then the problem (26) becomes equivalent to

$$\min_{M,\boldsymbol{\tau},\Upsilon\in\mathbb{R}^{t\times n}} \frac{1}{t}\sum_i [F(\tfrac{1}{t}X_{i:}\Upsilon'M+\boldsymbol{\tau}')-(\tfrac{1}{t}X_{i:}\Upsilon'M_{:i}+\tau_i)] + \frac{\gamma}{2}\operatorname{tr}(\Upsilon'M\Upsilon). \quad (28)$$

Note that $F = g$ can be interpreted as a soft max, hence the result is related to the typical max-margin style model. The loss of each example i is the soft max of $X_{i:}\Upsilon'M+\boldsymbol{\tau}'$ (a row vector) minus $X_{i:}\Upsilon'M_{:i}+\tau_i$. Here τ_i is an offset associated with each training example (cf. b_j for each cluster).

4.1 Optimization

The most straightforward method for optimizing (28) is to treat it as a convex function of M, whose gradient and objective value can be evaluated by minimizing out Υ and $\boldsymbol{\tau}$. Since both Υ and $\boldsymbol{\tau}$ are unconstrained, this can be easily accomplished by quasi-Newton methods like L-BFGS. Interestingly, thanks to the structure of the problem, we can optimize (28) even more efficiently by applying the same change of variable as in §3.2.1. Letting $V = M\Upsilon \in \mathbb{R}^{t\times n}$ and constraining M to $\mathcal{M}_3$, the problem (28) becomes

$$\min_{V,\boldsymbol{\tau}} \frac{\gamma}{2}\Omega^2(V)+\frac{1}{t}\sum_i [F(\tfrac{1}{t}X_{i:}V'+\boldsymbol{\tau}')-(\tfrac{1}{t}X_{i:}V'_{i:}+\tau_i)]. \quad (29)$$

This objective again absorbs the spectral constraints on M into the norm Ω, and can be readily solved by generalized conditional gradient in Algorithm 1. The extension to $M \in \mathcal{M}_2$ is also immediate.

5 Joint Generative Clustering

In all models considered so far, we have ignored the cluster prior $\mathbf{q}$. This quantity is often useful in practice for inference at the cluster level, and can often be effectively learned by joint generative models. In this section, we extend our convex relaxation technique to this setting.

Assume a multinomial distribution over cluster prior parameterized by $\mathbf{w} \in \mathbb{R}^d$: $p(\mathbf{Y} = j) = \exp(w_j - g(\mathbf{w}))$ where $g(\mathbf{w}) = \log\sum_i \exp(x_i)$. Then by (1) and (7), the negative log joint likelihood is: $-\mathbf{1}'Y\mathbf{w} + tg(\mathbf{w}) + L(YB) + \text{const}$. As above, one can add regularizers on $\mathbf{w}$ and B, as well as an entropic regularizer $h(Y'\mathbf{1})$ to encourage cluster diversity, yielding:

$$\min_{\mathbf{w},B,Y} -\frac{1}{t}\mathbf{1}'Y\mathbf{w}+g(\mathbf{w})+\frac{\beta}{2}\|Y\mathbf{w}\|^2+h(Y'\mathbf{1}) + \frac{1}{t}L(YB)+\frac{\alpha}{2}\|YB\|_F^2. \quad (30)$$

This formulation can be convexified in terms of M by using the same techniques as §4 and §3.2, respectively. In particular, consider the prior $p(Y)$ as a discriminative model $Z \to Y$, where Z can only take a constant scalar value 1. Then treating Z as the X in §4, it is easy to show that the first line of (30) can be relaxed into (ignoring the offset $\boldsymbol{\tau}$):

$$\min_{\mathbf{s}\in\mathbb{R}^t} \frac{\beta}{2}\operatorname{tr}(\mathbf{s}'M\mathbf{s})-\frac{1}{t}\mathbf{1}'M\mathbf{s}+g\left(\frac{1}{t}M\mathbf{s}\right). \quad (31)$$

Finally by applying the same technique that converted (14) to (15) in conditional model, one can reformulate (30) into:

$$\min_{A,M,\mathbf{s}} \frac{\beta}{2}\operatorname{tr}(\mathbf{s}'M\mathbf{s})-\frac{1}{t}\mathbf{1}'M\mathbf{s}+g(\tfrac{1}{t}M\mathbf{s}) + \frac{1}{t}L(MA)+\frac{\alpha}{2}\operatorname{tr}(A'MA). \quad (32)$$

To optimize this formulation, let $\mathbf{u} = M\mathbf{s} \in \mathbb{R}^t$ and $T =$

[3] Since $M^2 = M$ for $M \in \mathcal{M}$, (26) can also be recovered by setting $G(\Psi) = \frac{1}{2\gamma}\operatorname{tr}(\Psi\Psi')$. However, to reformulate the problem into (29), which is the key to efficient optimization, it is crucial to include $M^\dagger$ in G.

Data set	t	n	d	Data set	t	n	d
Yale	165	1024	15	Diabetes	768	8	2
ORL	400	1024	40	Heart	270	13	2
E-mail	1000	57	2	Breast	699	9	2
Balance	625	4	2				

Table 1: Properties of the data sets used in the experiments.

$MA \in \mathbb{R}^{t\times n}$. Then with $M \in \mathcal{M}_3$, (32) becomes

$$\min_{\mathbf{u},T} g\Big(\frac{\mathbf{u}}{t}\Big) - \frac{1}{t}\mathbf{1}'\mathbf{u} + \frac{1}{t}L(T) + \min_{M\in\mathcal{M}_3} \frac{\beta}{2}\mathbf{u}'M^{\dagger}\mathbf{u} + \frac{\alpha}{2}\mathrm{tr}(T'M^{\dagger}T)$$

$$= \min_{\mathbf{u},T} g\Big(\frac{\mathbf{u}}{t}\Big) - \frac{1}{t}\mathbf{1}'\mathbf{u} + \frac{1}{t}L(T) + \frac{1}{2}\Omega^2([\sqrt{\beta}\mathbf{u}, \sqrt{\alpha}T]), \quad (33)$$

which can be solved by the methods outlined above.

6 Experimental Evaluation

We evaluated the proposed convex relaxations for the three models developed in this paper: conditional (jointly convex or arbitrary Bregman divergence), joint, and discriminative.

Data sets. We used seven labeled data sets for these experiments. Five of them are from the UCI repository (Frank & Asuncion, 2010): Balance, Breast Cancer, Diabetes, Heart, and Spam E-mail. The two others are multiclass face data sets: ORL[4] and Yale[5]. We down-sampled Spam-Email to 1000 points while preserving the class ratio. The properties of these data sets are summarized in Table 1, giving the values of t, n, and d. We shifted all features to be nonnegative so that all transfer functions can be applied. Finally the features were normalized to unit variance.

Transfer functions. For all generative models, we tested two transfer functions: linear and sigmoid.

Parameters settings. To closely approximate the original objective without creating numerical difficulty, we chose all the regularization parameters α, β and γ to be reasonably small $\alpha \in \{10^{-5}, 10^{-9}\}$, $\beta \in \{10^{-5}, 10^{-9}\}$, $\gamma \in \{10^{-6}, 10^{-9}\}$ and report the experimental results for the choices that obtain highest accuracy. However, the results were not sensitive to these values.

6.1 Conditional: Jointly Convex Bregman Divergence

Algorithms. Our method (cvxCondJC) first minimizes $D_F(X, MX)$ as in (10), but over $M \in \mathcal{M}_1$. The optimal M is then rounded to a hard cluster assignment via spectral clustering (SC rounding, Shi & Malik, 2000). The result is further used to initialize a local re-optimization using the *original* objective $D_F(X, Y\Gamma)$. Since k-class spectral clustering involves a k-means algorithm, with random elements, this was repeated 10 times and variance reported.

[4]cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

[5]http://cvc.yale.edu/projects/yalefaces/yalefaces.html

	cvxCondJC +SC rounding	cvxCondJC +SC+re-opt	altCondJC
		Spam E-mail	
lin_obj($\times 10^2$)	9.4± 0.1	**9.3**± 0.0	**9.3**± 0.0
lin_acc(%)	71.5±11.6	**76.3**±13.6	75.1±12.6
sigm_obj($\times 10^3$)	7.8± 0.1	**7.7**± 0.1	**7.7**± 0.1
sigm_acc(%)	75.1±12.0	**80.0**± 9.4	76.0± 7.2
		ORL	
lin_obj($\times 10^3$)	3.3± 0.1	**2.0**± 0.0	2.1± 0.0
lin_acc(%)	**57.0**± 3.5	55.4± 2.9	40.6± 2.3
sigm_obj($\times 10^2$)	3.8± 0.1	**3.5**± 0.1	3.7± 0.1
sigm_acc(%)	57.8± 3.6	**58.2**± 4.1	48.2± 3.0
		Yale	
lin_obj($\times 10^1$)	5.6± 0.1	**5.5**± 0.0	5.8± 0.1
lin_acc(%)	46.8± 1.7	**47.0**± 2.1	44.5± 4.2
sigm_obj($\times 10^2$)	9.6± 0.4	**9.2**± 0.1	9.6± 0.3
sigm_acc(%)	49.9± 2.1	**51.5**± 2.1	46.6± 4.1
		Balance	
lin_obj($\times 10^1$)	7.2± 0.0	**7.1**± 0.0	7.2± 0.0
lin_acc(%)	57.1± 6.9	**57.3**± 7.1	54.2± 4.6
sigm_obj($\times 10^2$)	5.0± 0.3	**3.9**± 0.0	4.0± 0.0
sigm_acc(%)	49.3± 5.1	**50.5**± 5.1	49.4± 4.3
		Breast Cancer	
lin_obj($\times 10^2$)	1.8± 0.2	**1.6**± 0.0	1.7± 0.0
lin_acc(%)	72.5±12.7	**84.7**± 8.8	78.7±10.4
sigm_obj($\times 10^2$)	**8.5**± 0.2	**8.5**± 0.1	**8.5**± 0.1
sigm_acc(%)	72.4±13.7	**72.5**±13.7	70.6±11.6
		Diabetes	
lin_obj($\times 10^2$)	**2.0**± 0.1	**2.0**± 0.0	**2.0**± 0.0
lin_acc(%)	57.1± 0.5	**58.5**± 0.0	**58.5**± 0.1
sigm_obj($\times 10^3$)	1.2± 0.1	**1.1**± 0.0	**1.1**± 0.0
sigm_acc(%)	**58.8**± 3.9	58.2± 0.1	58.0± 0.6
		Heart	
lin_obj($\times 10^2$)	**1.3**± 0.0	**1.3**± 0.0	**1.3**± 0.0
lin_acc(%)	**68.1**±10.0	65.6± 7.8	65.4± 5.0
sigm_obj($\times 10^2$)	7.5± 0.2	**7.2**± 0.2	**7.2**± 0.2
sigm_acc(%)	63.4± 5.9	**64.9**± 6.6	64.4± 7.8

Table 2: Experimental results for the conditional model with jointly convex Bregman divergences. Here "lin" and "sigm" refer to linear and sigmoid transfers respectively. Best results in **bold**.

We compared our algorithm with altCondJC (hard EM), which optimizes $D_F(X, Y\Gamma)$ by alternating, with Y reinitialized randomly 30 times.

Results. In Table 2, the first and third rows of each block gives the optimal value of $D_F(X, Y\Gamma)$ found by altCondJC, and by cvxCondJC (both after SC rounding and re-optimization). The second and fourth lines give the highest accuracy among all possible matchings between the clusters and ground truth labels. Across all data sets and transfer functions, cvxCondJC with SC rounding and re-optimization finds a lower objective value and higher accuracy than altCondJC. In addition, although the objective

	cvxCond +SC rounding	cvxCond +SC rounding & re-opt	altCond
		Spam E-mail	
lin_obj($\times 10^2$)	**9.3**± 0.1	**9.3**± 0.0	**9.3**± 0.0
lin_acc(%)	75.0± 9.0	**79.8**±10.2	73.9±13.3
sigm_obj($\times 10^3$)	8.0± 0.2	**7.7**± 0.1	**7.7**± 0.1
sigm_acc(%)	64.8±12.5	**78.7**± 7.8	75.3± 5.5
		ORL	
lin_obj($\times 10^3$)	2.7± 0.1	**2.0**± 0.0	2.1± 0.0
lin_acc(%)	**62.6**± 3.0	59.4± 2.4	40.1± 2.3
sigm_obj($\times 10^2$)	4.0± 0.1	**3.4**± 0.0	3.7± 0.1
sigm_acc(%)	**60.1**± 6.1	60.0± 4.9	48.6± 2.7
		Yale	
lin_obj($\times 10^1$)	6.1± 0.2	**5.7**± 0.1	5.8± 0.1
lin_acc(%)	43.3± 3.2	**45.2**± 3.2	44.4± 4.0
sigm_obj($\times 10^2$)	10.3± 0.2	**9.3**± 0.1	9.5± 0.2
sigm_acc(%)	46.6± 2.6	**51.1**± 2.7	46.2± 3.0
		Balance	
lin_obj($\times 10^1$)	8.0± 0.4	**7.1**± 0.0	**7.1**± 0.0
lin_acc(%)	57.1± 6.9	**57.3**± 7.1	55.5± 5.1
sigm_obj($\times 10^2$)	4.0± 0.0	**3.9**± 0.0	4.0± 0.1
sigm_acc(%)	**54.1**± 8.3	53.0± 6.0	50.9± 5.2
		Breast Cancer	
lin_obj($\times 10^2$)	1.7± 0.1	**1.6**± 0.0	1.7± 0.0
lin_acc(%)	75.4±13.3	**85.8**± 6.6	78.7±10.9
sigm_obj($\times 10^2$)	8.8± 0.2	**8.5**± 0.1	8.6± 0.2
sigm_acc(%)	66.8± 8.4	**72.3**±12.5	70.3±11.0
		Diabetes	
lin_obj($\times 10^2$)	**2.0**± 0.0	**2.0**± 0.0	**2.0**± 0.0
lin_acc(%)	58.1± 0.6	**58.3**± 0.0	58.2± 0.1
sigm_obj($\times 10^3$)	1.2± 0.1	1.1± 0.0	**1.0**± 0.0
sigm_acc(%)	54.7± 3.0	**58.2**± 0.2	58.1± 0.5
		Heart	
lin_obj($\times 10^2$)	**1.3**± 0.0	**1.3**± 0.0	**1.3**± 0.0
lin_acc(%)	**69.4**± 9.3	67.0± 5.5	66.1± 5.2
sigm_obj($\times 10^2$)	7.2± 0.1	**7.1**± 0.1	7.3± 0.2
sigm_acc(%)	**66.9**±10.7	64.9± 8.2	65.8± 6.3

Table 3: Experimental results for the conditional model with arbitrary Bregman divergences. Best results shown in **bold**.

achieved after rounding might be higher than that of altCondJC, the accuracy is usually comparable. Overall, the final clustering found by cvxCondJC is superior to randomized local optimization.

6.2 Conditional: Arbitrary Bregman Divergence

Algorithms. Our method (cvxCond) first optimized (15) over $M \in \mathcal{M}_2$ using Algorithm 1. Then similar to §6.1, the optimal M was rounded by spectral clustering (10 repeats). Here subsequent re-optimization (based on local optimization) was performed on the objective $D_{F^*}(YB, f(X))$. The competing algorithm, altCond, optimizes this objective by alternating with 30 random initializations of Y.

	cvxDisc	JB	GS
		Spam E-mail	
run time ($\times 10^4$s)	**0.005**	0.651	2.148
obj w/ SC rounding ($\times 10^3$)	**8.0**± 0.2	8.7± 0.0	8.2± 0.2
obj w/ SC + re-opt ($\times 10^3$)	**7.6**± 0.0	7.9± 0.2	**7.6**± 0.0
acc w/ SC rounding (%)	**69.9**±14.3	60.7± 0.1	62.8± 9.2
acc w/ SC + re-opt (%)	**83.5**± 7.8	61.3± 9.2	81.4± 5.6
		ORL	
run time ($\times 10^4$s)	**0.080**	0.695	6.372
obj w/ SC rounding ($\times 10^2$)	4.1± 0.1	7.1± 0.0	**3.6**± 0.0
obj w/ SC + re-opt ($\times 10^3$)	**3.5**± 0.0	3.8± 0.1	3.6± 0.0
acc w/ SC rounding (%)	**59.4**± 2.7	20.0± 1.1	54.6± 2.1
acc w/ SC + re-opt (%)	**59.5**± 2.8	45.2± 2.5	54.6± 2.4
		Yale	
run time ($\times 10^3$s)	**0.050**	0.648	6.745
obj w/ SC rounding ($\times 10^3$)	**8.6**± 0.2	13.2± 0.0	10.2± 0.3
obj w/ SC + re-opt ($\times 10^3$)	**7.6**± 0.1	8.3± 0.1	7.8± 0.3
acc w/ SC rounding (%)	**44.3**± 2.5	16.2± 0.6	33.8± 3.6
acc w/ SC + re-opt (%)	**46.1**± 2.9	34.1± 2.6	42.4± 2.7
		Balance	
run time ($\times 10^4$s)	**0.004**	0.155	0.078
obj w/ SC rounding ($\times 10^2$)	5.1± 0.0	6.1± 0.0	**4.9**± 0.1
obj w/ SC + re-opt ($\times 10^2$)	**3.9**± 0.0	4.5± 0.0	4.1± 0.2
acc w/ SC rounding (%)	**62.0**± 2.3	47.0± 1.8	46.5± 6.3
acc w/ SC + re-opt (%)	58.7± 0.0	**62.3**± 1.8	52.2± 5.2
		Breast Cancer	
run time ($\times 10^4$s)	**0.006**	0.479	1.758
obj w/ SC rounding ($\times 10^2$)	**8.5**± 0.0	10.0± 0.0	9.1± 0.2
obj w/ SC + re-opt ($\times 10^2$)	**8.4**± 0.0	8.7± 0.3	**8.4**± 0.1
acc w/ SC rounding (%)	**79.8**±15.7	60.4± 3.6	72.3±10.3
acc w/ SC + re-opt (%)	80.7±12.5	60.0± 4.2	**84.4**± 8.8
		Diabetes	
run time ($\times 10^4$s)	**0.012**	1.722	2.731
obj w/ SC rounding ($\times 10^3$)	**1.2**± 0.1	1.4± 0.0	1.3± 0.1
obj w/ SC + re-opt ($\times 10^3$)	**1.1**± 0.0	**1.1**± 0.0	**1.1**± 0.0
acc w/ SC rounding (%)	53.5± 3.1	**64.8**± 0.0	56.6± 4.2
acc w/ SC + re-opt (%)	58.3± 0.2	**58.6**± 0.0	58.3± 0.2
		Heart	
run time ($\times 10^4$s)	**0.001**	0.212	6.848
obj w/ SC rounding ($\times 10^2$)	**7.6**± 0.4	8.6± 0.0	7.7± 0.4
obj w/ SC + re-opt ($\times 10^3$)	**7.3**± 0.3	7.9± 0.0	**7.3**± 0.2
acc w/ SC rounding (%)	61.7± 5.8	55.2± 0.0	**64.4**± 9.5
acc w/ SC + re-opt (%)	**66.0**± 5.7	51.1± 0.0	65.2± 8.4

Table 4: Experimental results for the discriminative models.

Results. The results in Table 3 are organized in the same manner as Table 2. Here it can be observed that for all data sets and transfer functions, cvxCond with SC rounding and reoptimization yields lower optimal objective value and higher accuracy than altCond (except Diabetes/sigm). Moreover, the objective values also exhibits lower standard deviation than altCond, which suggests that the value regularization scheme helps stabilize the reoptimization. Finally note the accuracy of cvxCond with rounding is already comparable with that of altCond on most data sets.

6.3 Discriminative Models

Algorithms. Our method (cvxDisc) optimized (28) over $M \in \mathcal{M}_2$ by solving (29). We also tested on the algorithms of (Joulin & Bach, 2012) and (Guo & Schuurmans, 2007), which we refer to as JB and GS respectively. The result of all the three methods were rounded by spectral clustering, then used to initialize a local re-optimization over $D_F(X, Y\Gamma)$. Since the discriminative model is logistic, we used the sigmoid transfer in D_F only.

Results. According to Table 4, it is clear that even without reoptimization, cvxDisc after rounding already achieves higher or comparable accuracy to both JB and GS in all cases. Further improvements are obtained by reoptimization. Regarding the run time for solving the respective convex relaxations, cvxDisc is at least 10 times faster than both JB and GS. This confirms the computational advantage of our primal reformulation (28), compared to other implementations of convex relaxation.

6.4 Joint Generative Models

Algorithms. Our proposed method, cvxJoint, optimizes (32) over $M \in \mathcal{M}_2$ by solving (33). As before, we rounded the optimal M by spectral clustering, and used the Y to initialize local reoptimization of the joint likelihood $-\mathbf{1}'Y\mathbf{w} + tg(\mathbf{w}) + L(YB)$.

We compared the results to those of three soft generative models. The standard soft EM (Banerjee et al., 2005, Algorithm 3) was randomly reinitialized 20 times. The other two algorithms are LG (Lashkari & Golland, 2007), and NB[6] (Nowozin & Bakir, 2008). Since they do not directly control the number of clusters, we tuned their parameters so that the resulting number of cluster is d, or a little higher than d which could be truncated based on the cluster prior.

Results. Since joint models also learn a cluster prior, accuracy can take two forms. The hard accuracy is computed by $\text{argmax}_y\, p(y|\mathbf{x}_i) = \text{argmax}_y\, p(y)p(\mathbf{x}_i|y)$ in the case of soft EM, LG, and NB. Our model outputs a hard accuracy by locally reoptimizing the joint likelihood. For all methods, we define the soft accuracy based on the posterior distribution: $\max_\pi \mathbb{E}_{Y\sim p(Y|X)}[\text{Accuracy}(Y, \pi(Y^*))]$, where Y^* is the ground truth label and π is a matching between the cluster and label.

As can be observed from Table 5, cvxJoint with rounding and reoptimization achieves superior or comparable performance to the competing algorithms in most cases (except three settings in Balanced and one each in Yale and Diabetes), both in terms of hard *and* soft accuracy.

	linear		sigmoid	
	acc(%)	soft acc(%)	acc(%)	soft acc(%)
	Spam E-mail			
cvxJoint1	55.7±1.9	55.9±1.4	62.6±9.0	67.7±11.0
cvxJoint2	**60.5**±0.0	**60.5**±0.0	**81.5**±16.4	**79.2**±15.1
softEM	**60.5**±0.0	54.5±2.6	58.2±7.4	52.9±2.0
LG	60.0	0.1	40.6	1.8
NB	**60.5**	51.4	NA	NA
	ORL			
cvxJoint1	**61.0**±1.3	52.6±1.5	**63.0**±2.3	58.6±1.8
cvxJoint2	55.9±1.4	**52.8**±1.2	58.7±2.7	**58.7**±2.7
softEM	39.6±2.1	37.0±2.0	44.9±3.1	44.7±3.1
LG	40.0	1.9	36.0	0.5
NB	12.0	5.3	NA	NA
	Yale			
cvxJoint1	**47.9**±3.8	**45.9**±3.1	61.9±8.3	55.9±1.4
cvxJoint2	45.8±3.4	45.1±3.1	60.5±0.0	**60.5**±0.0
softEM	39.6±2.1	37.0±2.0	60.5±0.0	**60.5**±0.0
LG	35.2	4.8	**66.9**	0.1
NB	20.6	10.4	NA	NA
	Balance			
cvxJoint1	50.5±2.3	36.3±0.7	51.6±2.7	39.5±1.2
cvxJoint2	46.1±0.0	46.1±0.0	46.1±0.0	**46.1**±0.0
softEM	46.1±0.0	38.1±2.8	46.1±0.0	39.6±0.0
LG	**57.4**	0.2	**59.0**	0.2
NB	54.2	**54.7**	NA	NA
	Breast Cancer			
cvxJoint1	**71.0**±11.9	56.9±4.7	**70.9**±13.0	63.9±8.1
cvxJoint2	65.5±0.0	**65.5**±0.0	65.5±0.0	**65.5**±0.0
softEM	65.5±0.0	57.7±4.5	65.5±0.0	55.5±5.4
LG	61.8	0.1	65.5	0.1
NB	69.8	50.3	NA	NA
	Diabetes			
cvxJoint1	56.0±2.6	53.6±2.5	57.5±5.5	57.6±5.6
cvxJoint2	**65.1**±0.0	**65.1**±0.0	62.0±3.3	**62.6**±2.6
softEM	**65.1**±0.00	57.6±4.6	**65.1**±0.0	57.4±5.2
LG	56.8	0.1	58.5	0.1
NB	**65.1**	60.2	NA	NA
	Heart			
cvxJoint1	**63.0**±6.4	53.3±1.8	63.0±7.4	61.0±6.2
cvxJoint2	55.6±0.0	**55.5**±0.0	**64.0**±7.5	**61.3**±7.1
softEM	55.6±0.0	51.7±1.6	55.6±0.0	52.7±0.0
LG	57.4	0.4	55.2	0.4
NB	55.6	53.0	NA	NA

Table 5: Experimental results for the joint generative model. Here cvxJoint1 is cvxJoint followed by SC rounding, whereas cvxJoint2 uses additional re-optimization. Best results in **bold**.

7 Conclusion

In this paper we constructed convex relaxations for clustering with Bregman divergences. Using normalized equivalence relations, we also designed efficient algorithms for optimizing the models. For future work, it will be interesting to extend these approaches to generative soft clustering, and further scale up the optimization to large applications.

Acknowledgements

This research is supported by AICML and NSERC. We thank Junfeng Wen and Yaoliang Yu for their helpful discussions and early assistance. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

[6] http://www.nowozin.net/sebastian/infex. Since their approach relies heavily on the Gaussian model, we put NA in the corresponding cells in Table 5.

References

Aloise, D., Seshpande, A., Hansen, P., and Popat, P. Np-hardness of Euclidean sum-of-squares clustering. *Machine Learning*, 75:245–249, 2009.

Anandkumar, A., Hsu, D., and Kakade, S. A method of moments for mixture models and hidden Markov models. In *Proc. Conference on Learning Theory*, 2012.

Arora, S. and Kannan, R. Learning mixtures of separated nonspherical Gaussians. *The Annals of Applied Probability*, 15 (1A):69–92, 2005.

Bach, F. and Harchaoui, Z. Diffrac: A discriminative and flexible framework for clustering. In *Advances in Neural Information Processing Systems 20*, 2007.

Banerjee, A., Merugu, S., Dhillon, I. S., and Ghosh, J. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

Berman, A. and Xu, C. 5×5 completely positive matrices. *Linear Algebra and its Applications*, 393:55–71, 2004.

Borwein, J. and Lewis, A. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. CMS books in Mathematics. Canadian Mathematical Society, 2000.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–123, 2010.

Chapelle, O., Schölkopf, B., and Zien, A. (eds.). *Semi-Supervised Learning*. MIT Press, 2006.

Chaudhuri, K., Dasgupta, S., and Vattani, A. Learning mixtures of Gaussians using the k-means algorithm. arXiv:0912.0086v1, 2009.

Dasgupta, S. The hardness of k-means clustering. Technical Report CS2008-0916, CSE Department, UCSD, 2008.

Dasgupta, S. and Schulman, L. A probabilistic analysis of EM for mixtures of separated, spherical Gaussians. *Journal of Machine Learning Research*, 8:203–226, 2007.

Dempster, A., Laird, N., and Rubin, D. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–22, 1977.

Frank, A. and Asuncion, A. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Guo, Y. and Schuurmans, D. Convex relaxations of latent variable training. In *Adv. Neural Infor. Processing Systems 20*, 2007.

Hansen, P., Jaumard, B., and Mladenovic, N. Minimum sum of squares clustering in a low dimensional space. *Journal of Classification*, 15(1):37–55, 1998.

Horn, R. and Johnson, C. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985.

Hsu, D. and Kakade, S. Learning mixtures of spherical Gaussians: Moment methods and spectral decompositions. In *Innovations in Theoretical Computer Science (ITCS)*, 2013.

Inaba, M., Katoh, N., and Imai, H. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering. In *Proc. Symp. Computational Geometry*, 1994.

Joulin, A. and Bach, F. A convex relaxation for weakly supervised classifiers. In *Proceedings of the International Conference on Machine Learning*, 2012.

Joulin, A., Bach, F., and Ponce, J. Efficient optimization for discriminative latent class models. In *Advances in Neural Information Processing Systems 23*, 2010.

Kalai, A., Moitra, A., and Valiant, G. Efficiently learning mixtures of two Gaussians. In *Proceedings ACM Symposium on Theory of Computing*, 2010.

Kumar, A., Sabharwal, Y., and Sen, S. A simple linear time $(1+\epsilon)$-approximation algorithm for k-means clustering in any dimensions. In *Proc. Symposium on Foundations of Computer Science,*, 2004.

Lashkari, D. and Golland, P. Convex clustering with exemplar-based models. In *Advances in Neural Information Processing Systems 20*, 2007.

Laue, S. A hybrid algorithm for convex semidefinite optimization. In *Proceedings of the International Conference on Machine Learning*, 2012.

MacQueen, J. Some methods of classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symposium on Math., Stat., and Prob.*, pp. 281. 1967.

Mirsky, L. *An Introduction to Linear Algebra*. Oxford, 1955.

Mirsky, L. A trace inequality of John von Neumann. *Monatsh. Math.*, 79(4):303–306, 1975.

Moitra, A. and Valiant, G. Settling the polynomial learnability of mixtures of Gaussians. In *Proc. Symposium on Foundations of Computer Science,*, 2010.

Neal, R. and Hinton, G. A view of the EM algorithm that justifies incremental, sparse, and other variants. In Jordan, M. (ed.), *Learning in Graphical Models*. Kluwer, 1998.

Ng, A., Jordan, M., and Weiss, Y. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, 2001.

Nowozin, S. and Bakir, G. A decoupled approach to exemplar-based unsupervised learning. In *Proceedings of the International Conference on Machine Learning*, 2008.

Peng, J. and Wei, Y. Approximating k-means-type clustering via semidefinite programming. *SIAM J. on Optimization*, 18:186–205, 2007.

Shi, J. and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

Tsuda, K., Rätsch, G., and Warmuth, M. Matrix exponentiated gradient updates for on-line learning and Bregman projections. In *Advances in Neural Information Processing Systems 17*, 2004.

Wang, S. and Schuurmans, D. Learning continuous latent variable models with Bregman divergence. In *International Conference on Algorithmic Learning Theory*, 2003.

Xing, E. and Jordan, M. On semidefinite relaxation for normalized k-cut and connections to spectral clustering. Technical Report UCB/CSD-03-1265, EECS Department, University of California, Berkeley, 2003.

Xu, L. and Schuurmans, D. Unsupervised and semi-supervised multi-class support vector machines. In *Proc. Conf. Association for the Advancement of Artificial Intelligence (AAAI)*, 2005.

Zass, R. and Shashua, A. A unifying approach to hard and probabilistic clustering. In *Proc. Intl. Conf. Computer Vision*, 2005.

Zha, H., Ding, C., Gu, M., He, X., and Simon, H. Spectral relaxation for k-means clustering. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

Zhang, X., Yu, Y., and Schuurmans, D. Accelerated training for matrix-norm regularization: A boosting approach. In *Advances in Neural Information Processing Systems 25*, 2012.

Learning Sparse Causal Models is not NP-hard

Tom Claassen, Joris M. Mooij, and Tom Heskes
Institute for Computer and Information Science
Radboud University Nijmegen
The Netherlands

Abstract

This paper shows that causal model discovery is *not* an NP-hard problem, in the sense that for sparse graphs bounded by node degree k the sound and complete causal model can be obtained in worst case order $N^{2(k+2)}$ independence tests, even when latent variables and selection bias may be present. We present a modification of the well-known FCI algorithm that implements the method for an independence oracle, and suggest improvements for sample/real-world data versions. It does not contradict any known hardness results, and does not solve an NP-hard problem: it just proves that sparse causal discovery is perhaps more complicated, but not as *hard* as learning minimal Bayesian networks.

1 Introduction

Causal discovery is one of the cornerstones behind scientific progress. In recent years, significant breakthroughs have been made in causal inference under very reasonable assumptions, even when only data from observations are available (Pearl, 2000; Spirtes et al., 2000). Still, it is probably safe to say that many researchers consider causal discovery to be a difficult problem, and that it is generally thought to be computationally at least as hard as related problems such as learning minimal Bayesian networks.

The class of NP problems (non-deterministic, polynomial time) are problems for which a solution can be *verified* in polynomial time: order $O(N^k)$ for some constant k given input size N, but for which no known polynomial time algorithm exists (or is thought to exist) that *finds* such a solution. In many cases algorithms exist that are able to find a solution quickly, or at least provide good approximations, but still have worst-case exponential running time order $O(2^{N^k})$. Typical examples include finding the shortest path through all nodes in a weighted graph (travelling salesman), coloring vertices in a graph so that no two adjacent vertices share the same color, probabilistic inference in a Bayesian network (Cooper, 1990), and the Boolean satisfiability problem (k-SAT, the first known NP-complete problem).

A problem is NP-hard if it is at least as hard as the hardest problems in NP. Put differently, a problem is NP-hard if there is a polynomial time reduction of an NP-complete problem to it, so that any polynomial time solution to the NP-hard problem implies that all NP-complete problems can be solved in polynomial time. See (Garey and Johnson, 1979) for a standard introduction to the subject, and e.g. (Goldreich, 2008) for a more modern approach.

In this article we focus on the problem of learning a causal model from probabilistic information. We assume there is a system that is characterized by a so-called causal DAG $\mathcal{G}_C$ that describes the causal interactions between the variables in the system. The structure of this causal DAG is invariant and responsible for a probability distribution over the subset of observed variables. The goal is to learn as much as possible about the presence or absence of certain causal relations between variables in the underlying causal DAG from the available probabilistic information.

Chickering et al. (2004) showed that finding an inclusion-optimal (minimal) Bayesian network for a given probability distribution is NP-hard, even when a constant-time independence oracle is available (see §2 for more details). Such hardness results have inspired many creative approaches to network learning, e.g. methods that seek to find efficient approximations to minimal Bayesian networks through greedy search (Chickering, 2002), or methods that employ specialized heuristics or solver techniques to make exact learning feasible from 30, up to even 60 variables if the graph is sufficiently sparse (Yuan and Malone, 2012; Cussens, 2011).

On the face of it, minimal Bayesian network inference seems close to a simplified and idealized version of a causal model discovery problem. Hence it may be unsurprising to find that currently available methods also have worst-case exponential complexity (see §3.2).

However, in this paper we show that causal model discovery in sparse graphs is in fact *not* an NP-hard problem, even in the presence of latent confounders and/or selection bias. We do this by providing an adaptation of the well-known FCI algorithm (see §3.2) for learning the sound and complete causal model in the form of a PAG with running time in the worst case polynomial in the number of independence tests to order $O(N^{2(k+2)})$, where N is the number of variables and k the maximum node degree in the causal model over the observed variables.

Graphical causal models

A *mixed graph* $\mathcal{G}$ is a graphical model that can contain three types of edges between pairs of nodes: directed ($\rightarrow$), bi-directed ($\leftrightarrow$), and undirected ($-$). In a mixed graph, standard graph-theoretical notions, e.g. *child/parent*, *ancestor/descendant*, *directed path, cycle*, still apply, with natural extension to sets. In particular $Adj_{\mathcal{G}}(\mathbf{X})$ indicates all nodes adjacent to (but not in) the set of nodes $\mathbf{X}$ in the graph $\mathcal{G}$, and $An_{\mathcal{G}}(\mathbf{X})$ indicates all (ancestors of) nodes in $\mathbf{X}$ in the graph $\mathcal{G}$. A vertex Z is a *collider* on a path $u = \langle \ldots, X, Z, Y, \ldots \rangle$ if there are arrowheads at Z on both edges from X and Y, otherwise it is a *noncollider*.

A mixed graph $\mathcal{G}$ is *ancestral* iff an arrowhead at X on an edge to Y implies there is no directed path from X to Y in $\mathcal{G}$, and there are no arrowheads at nodes with undirected edges. As a result, arrowhead marks can be read as 'is not an ancestor of'. In a mixed graph $\mathcal{G}$, a vertex X is *m-connected* to Y by a path u, relative to a set of vertices $\mathbf{Z}$, iff every noncollider on u is not in $\mathbf{Z}$, and every collider on u is an ancestor of $\mathbf{Z}$. If there is no such path, then X and Y are *m-separated* by $\mathbf{Z}$. An ancestral graph is *maximal* (MAG) if for any two nonadjacent vertices there is a set that separates them. A *directed acyclic graph* (DAG) is a special kind of MAG, containing only $\rightarrow$ edges, for which m-separation reduces to the familiar d-separation criterion. The *Markov property* links the structure of an ancestral graph $\mathcal{G}$ to observed probabilistic independencies: $X \perp\!\!\!\perp Y \,|\, \mathbf{Z}$, if X and Y are m-separated by $\mathbf{Z}$. *Faithfulness* implies that the only observed independencies in a system are those entailed by the Markov property. For more details, see (Koller and Friedman, 2009; Spirtes et al., 2000).

A *causal DAG* $\mathcal{G}_C$ is a directed acyclic model where the arcs represent direct causal interactions (Pearl, 2000). In general, the independence relations between observed variables in a causal DAG can be represented in the form of a MAG (Richardson and Spirtes, 2002). The (complete) partial ancestral graph (PAG) represents all invariant features that characterize the equivalence class $[\mathcal{G}]$ of such a MAG, with a tail '$-$' or arrowhead '$>$' end mark on an edge, iff it is invariant in $[\mathcal{G}]$, otherwise it has a circle mark '$\circ$', see (Zhang, 2008). Figure 1 illustrates the relation between these three types of graphs.

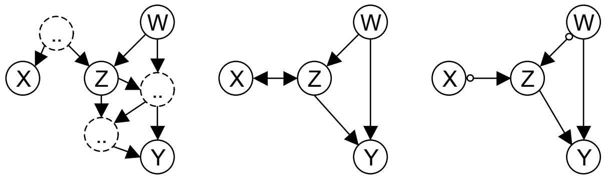


Figure 1: (a) Assumed underlying causal DAG $\mathcal{G}_C$; (b) corresponding MAG $\mathcal{M}$ over observed variables; (c) causal model as PAG, with $N = 4$ and $k \leq 3$.

C-LEARN

The completed PAG represents all valid causal information that can be inferred from independencies between observed variables: this will be the target causal model of the learning task (C-LEARN).

To distinguish between the contribution of the learning task and the calculation of the independencies themselves, we introduce the following notion from (Chickering et al., 2004):

Definition 1. An **independence oracle** for a distribution $p(\mathbf{X})$ is an oracle that, in constant time, can determine whether or not $X \perp\!\!\!\perp Y \,|\, \mathbf{Z}$, for any $(\{X, Y\} \cup \mathbf{Z}) \subseteq \mathbf{X}$.

The **C-LEARN** task can now be described as:
INSTANCE: Given an independence oracle $\mathcal{O}$ for a set of variables $\mathbf{X} = \{X_1, .., X_N\}$ that is faithful to an underlying causal DAG $\mathcal{G}_C$, and constant bound k.
TASK: Find the completed PAG model that matches $\mathcal{O}$ with node degree $\leq k$.

2 Why learning minimal Bayesian networks is NP-hard

It has long been known that learning a minimal Bayesian network is an NP-hard problem, even when an independence oracle is available and each node in the network has at most $k \geq 3$ parents. An elegant proof is provided in (Chickering et al., 2004) by introducing a polynomial reduction from an established NP-complete problem known as *Degree-Bounded Feedback Arc Set* (DBFAS) to an instance of learning a

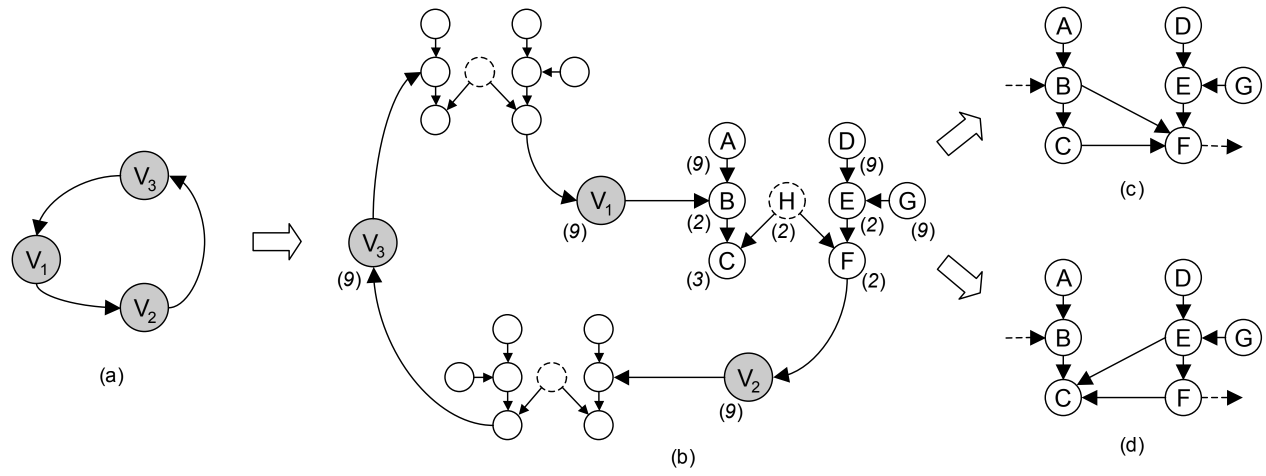


Figure 2: (a) DBFAS instance over 3 nodes; (b) corresponding Bayesian network reduction with distinct variables $\{A, B, C, D, E, F, G, H\}$ for each edge-component (nr. of states per variable in brackets below); (c) BN edge-component configuration with 16 parameters to specify the distribution over C and F; (d) idem, with 18 parameters.

minimal Bayesian network (B-LEARN) with maximum node degree k. Given an arbitrary network of directed edges the target in DBFAS is to find the smallest set of arcs (the 'feedback set') whose removal eliminates all directed cycles from the graph.

The reduction strategy from DBFAS to B-LEARN is depicted in Figure 2. It replaces the DBFAS instance with an equivalent Bayesian network, in which each directed edge $V_i \rightarrow V_j$ in DBFAS is replaced by an edge-component over discrete variables $\{A, B, C, D, E, F, G, H\}$. All DBFAS nodes V_i and all nodes $\{A, D, G\}$ in each edge-component get 9 states, the nodes $\{B, E, F, H\}$ get 2 states, and only the nodes C get 3 states. Furthermore, the H nodes in each edge-component are supposedly hidden, i.e. not in the set of observed variables in the minimal Bayesian network from B-LEARN, see Figure 2.

The connection between the instance of DBFAS and B-LEARN relies on the fact that it is *impossible* to construct a Bayesian network that can faithfully represent the independence relations from the underlying causal DAG when the H variables are not observed. As a result, each edge component is forced to choose between configuration (c) or (d) in Figure 2, with a preference for the smaller (c), *unless* this introduces a directed cycle due to the connection via $V_i \rightarrow B \rightarrow F \rightarrow V_j$. In that case the cycle can be broken by opting for (d). The (acyclic) minimal global Bayesian network will have the fewest edge-components oriented as (d) needed to break all cycles, and so all these edge components together represent a *minimal feedback arc set* for the original DBFAS problem.

The transformation from DBFAS to B-LEARN is polynomial, which implies that if there is any method that can solve B-LEARN in polynomial time, then it can also solve DBFAS in polynomial time, and with it the entire class of NP-complete problems. In particular, it also applies when there is a constant time independence oracle for B-LEARN and the max. node degree in the optimal solution is bounded by any $k \geq 3$. It is exactly this subclass of problems that are the focus of this paper in the context of learning a causal model.

Causal model reduction

An obvious question would be why the previous result should not apply to causal models in general, especially given the importance, explicit or implicit, of minimality in many structure learning tasks.

The answer to this lies in the fact that the 'Bayesian network' requirement itself imposes the additional restriction on the solution that only directed edges are allowed, which does not come into play when causal models are concerned. Therefore B-LEARN cannot opt for the configuration depicted in Figure 3, due to the edge $C \leftrightarrow F$, whereas C-LEARN has no such problem. This structure is actually *smaller* than the corre-

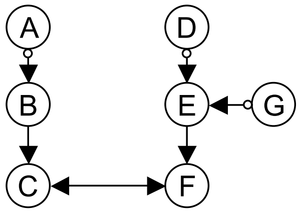


Figure 3: Minimal edge-component configuration in causal model, requiring 12 free parameters for variables C and F.

sponding minimal Bayesian network component, as it requires only 12 free parameters to specify the contribution of each $C-F$ pair in the network: $2\times2\times2=8$ for the C node, $2\times2\times1=4$ for F, and 1 for the H node, giving a total of 13 parameters, one of which can be marginalized out, see, e.g. (Evans and Richardson, 2010) for details on parameterizing acyclic directed mixed graphs (ADMGs), allowing for both directed and bi-directed edges.

As a result, a DBFAS reduction to C-LEARN along the lines of (Chickering et al., 2004) will fail to solve the original NP-complete problem, as the causal model will have *all* edge components oriented as in Figure 3, giving no information whatsoever on a minimal feedback arc set for the corresponding DBFAS instance.

We now turn to existing methods for causal discovery to see where they run into exponential trouble.

3 Independence based Causal Discovery

3.1 Causally sufficient systems

Many methods have been developed that efficiently learn the correct causal model when there are no unobserved common causes between the variables, e.g. IC (Pearl and Verma, 1991), PC (Spirtes et al., 2000), Grow-Shrink (Margaritis and Thrun, 1999), etc.

As a typical example, a version of PC is depicted in Algorithm 1. From a fully connected undirected graph $\mathcal{G}$, it consists of two stages: an adjacency search to remove edges, followed by an orientation phase. In the first stage, for each pair of nodes $X-Y$ (still) connected in $\mathcal{G}$ it searches for a subset of adjacent nodes $\mathbf{Z}$ that can separate them: $X \perp\!\!\!\perp Y \,|\, \mathbf{Z}$; if found the edge is removed. By checking all adjacent node-pairs in $\mathcal{G}$ for possible separating sets of increasing size, the PC algorithm ensures that it finds separating sets as small as possible. If the node degree in the true causal model $\mathcal{G}$ is bounded by k, then worst case it needs to check all subsets size $1..k$ from N nodes, for N^2 nodes on edges, resulting in a polynomial running time dominated by order $O(N^{k+2})$ for the adjacency search. Afterwards an orientation phase adds all invariant edge-marks (tails or arrowheads) by rules that trigger on the existence of certain path-configurations in $\mathcal{G}$, which can be checked in order $O(N^3)$ (Spirtes et al., 2000).

3.2 Causal discovery with latent confounders

Unfortunately, the PC algorithm can run into trouble when applied to causal models where causal sufficiency is not guaranteed. In that case it can miss certain separating sets that may require nodes not adjacent to either of the separated nodes. Figure 4(b) depicts the canonical 5-node example, where the edge $X-Y$ can be eliminated by the set $\{U,V,Z\}$, but this is not found by the PC algorithm, as at that stage Z is no longer adjacent to X or Y. As a result, edges may fail to be eliminated from $\mathcal{G}$, possibly leading to erroneous causal conclusions in the orientation stage, see e.g. (Colombo et al., 2012, §3) for more examples.

Algorithm 1 PC-algorithm

```
    In : independence oracle $\mathcal{O}$ for variables $\mathbf{V}$
    Out: causal model $\mathcal{G}$ over $\mathbf{V}$
    Adjacency search
 1: $\mathcal{G} \leftarrow$ fully connected undirected graph over $\mathbf{V}$
 2: $n = 0$
 3: repeat
 4:     repeat
 5:         select $X$ with $|Adj_{\mathcal{G}}(X)| > n$,
 6:         select $Y \in Adj_{\mathcal{G}}(X)$
 7:         repeat
 8:             select subset $\mathbf{Z}$ size $n$ from $Adj_{\mathcal{G}}(X) \setminus Y$,
 9:             if $X \perp\!\!\!\perp Y \,|\, \mathbf{Z}$ then
10:                 $Sepset(X,Y) = Sepset(Y,X) = \mathbf{Z}$
11:                 remove edge $X-Y$ from $\mathcal{G}$
12:             end if
13:         until all subsets size $n$ have been tested
14:     until all edges $X-Y$ in $\mathcal{G}$ have been checked
15:     $n = n+1$
16: until no more nodes with $|Adj_{\mathcal{G}}(X)| > n$
    Orientation phase
17: for all unshielded triples $X-Z-Y$ in $\mathcal{G}$ do
18:     if $Z \notin Sepset(X,Y)$ then
19:         orient v-structure $X \rightarrow Z \leftarrow Y$
20: end for
21: run other orientation rules until no more new
22: return causal model $\mathcal{G}$
```

To tackle this problem, Spirtes et al. (1999) developed the so-called Fast Causal Inference (FCI) algorithm that introduces an additional stage to the adjacency search. It searches for an extended set of nodes: the **D-SEP**(A,B) set, roughly corresponding to ancestors of $\{A,B\}$ that are adjacent to A and/or reachable via a bi-directed path, for which it can be shown that (with observed variables $\mathbf{O}$ and selection set $\mathbf{S}$):

Lemma 1. (Spirtes et al., 1999, Lemma 12) If there is some subset $\mathbf{W} \subseteq \mathbf{O} \setminus \{A,B\}$ such that A and B are d-separated by $\mathbf{W} \cup \mathbf{S}$, then A and B are d-separated given **D-SEP**$(A,B) \cup \mathbf{S}$.

The problem is that at that stage it is not yet known which nodes exactly belong to $An(\{A,B\})$. The solution employed by FCI is as follows: after the initial PC-adjacency search it adds some orientation informa-

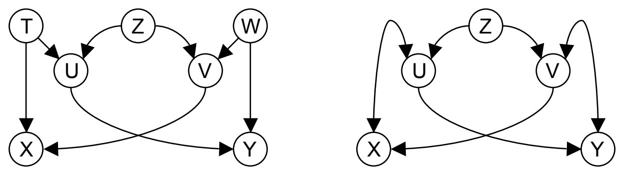


Figure 4: (a) With causal sufficiency (no confounders) we only need to check subsets from $Adj(X)$ or $Adj(Y)$, e.g. $X \perp\!\!\!\perp Y \mid V, T$; (b) Without causal sufficiency node $Z \notin Adj(X, Y)$ is needed in the separating set: $X \perp\!\!\!\perp Y \mid U, V, Z$

tion to the resulting adjacency graph to obtain a partially oriented graph π_0, and from this identifies the set **Possible-D-SEP**(A, B, π_0) for edge $A - B$ that is a guaranteed superset of **D-SEP**(A, B). It then tests for all subsets of **Possible-D-SEP**(A, B, π_0) until a separating set is found, or not, in which case the edge should remain present.

But even for graphs with degree $\leq k$, neither the size of the **Possible-D-SEP**(A, B, π_0) set, nor the size of the target **D-SEP** set is bounded by k. As a result, the worst case is now dominated by a term that could involve searching through all subsets from N nodes, which brings it back to exponential order time complexity $O(2^N)$.

In practice, the number of edges removed in the FCI stage tends to be very small (relative to the PC-stage), even though it is often the most expensive part of the algorithm by far. This behaviour is exploited effectively in the RFCI algorithm (Colombo et al., 2012): it skips the additional FCI stage and ensures the output is still sound (though no longer necessarily complete) through a modified orientation phase that avoids some of the erroneous causal conclusions PC could make when edges are missed.

In this article we take a different approach by showing that it is possible to build up an alternative **D-SEP** set that is completely determined by nodes adjacent to $\{A, B\}$ and already inferred minimal separating sets from the PC-adjacency search.

4 *D*-separating sets

First some terminology for the target nodes and edges:

Definition 2. In a MAG $\mathcal{M}$, two nodes X and Y are ***D*-separated** by $\mathbf{Z}$ iff they are d-separated by $\mathbf{Z}$, and all sets that *can* separate X and Y contain at least one node $Z \notin Adj(\{X, Y\})$. Such a node $Z \in \mathbf{Z}$ that cannot be made redundant by nodes adjacent to X or Y is a ***D*-sep node**, and (X, Y) is a ***D*-sep link**,

For example in Figure 4(b), X and Y are D-separated by $\{U, V, Z\}$, including D-sep node Z.

After the PC-adjacency search all D-sep links are still connected by an edge in the skeleton $\mathcal{G}$. Below we first discuss how to recognize possible D-sep links, and then how to find an appropriate D-separating set, including the elusive D-sep nodes. The third part puts it all together into a sound and complete search strategy. Proof details can be found in the Appendix and in (Claassen et al., 2013).

4.1 Identifying *D*-sep links

To characterize D-separable nodes, we use a result from (Spirtes et al., 1999; Claassen and Heskes, 2011):

Lemma 2. For disjoint (subsets of) nodes $X, Y, Z, \mathbf{Z}$ from the observed variables $\mathbf{O}$ in a causal graph $\mathcal{G}$ with selection set $\mathbf{S}$,

(1) $X \not\perp\!\!\!\perp Y \mid \mathbf{Z} \cup [Z] \quad \Rightarrow \quad Z \notin An_{\mathcal{G}}(\{X, Y\} \cup \mathbf{Z} \cup \mathbf{S})$.

(2) $X \perp\!\!\!\perp Y \mid [\mathbf{Z} \cup Z] \quad \Rightarrow \quad Z \in An_{\mathcal{G}}(\{X, Y\} \cup \mathbf{S})$,

where square brackets indicate a *minimal* set of nodes.

Rule (1) identifies invariant arrowheads on edges; rule (2) is used in §4.2 to build up a D-separating set. With Lemma 2 it is easy to show the following properties:

Lemma 3. In a MAG $\mathcal{M}$, if two nodes X and Y are D-separated by a minimal set $\mathbf{Z}$, then:

1. $X \notin An(\{Y\} \cup \mathbf{Z} \cup \mathbf{S})$
2. $Y \notin An(\{X\} \cup \mathbf{Z} \cup \mathbf{S})$
3. $\forall Z \in \mathbf{Z} : Z \in An(\{X, Y\} \cup \mathbf{S})$

It motivates the following introduction:

Definition 3. The **Augmented Skeleton** $\mathcal{G}^+$ is obtained from the skeleton $\mathcal{G}$ by adding all invariant arrowheads that follow from single node minimal dependencies $X \not\perp\!\!\!\perp Y \mid \mathbf{Z} \cup [W]$ by Lemma 2, rule(1).

By testing for dependence on adding single nodes to the minimal separating sets from the PC-adjacency search all invariant arrowheads in $\mathcal{G}^+$ are found. Note that only nodes Z adjacent to $\{X, Y\} \cup \mathbf{Z}$ in $\mathcal{G}$ need to be tested, and that all arrowheads from FCI's partially oriented graph π_0 (see §3.2) are also in $\mathcal{G}^+$.

The important implication is that D-sep links take on a very distinct pattern in $\mathcal{G}^+$:

Lemma 4. For a MAG $\mathcal{M}$, let X and Y be D-separable nodes that are adjacent in the corresponding augmented skeleton $\mathcal{G}^+$. If there are no edges in $\mathcal{G}^+$ between (other) D-separable nodes in $An(\{X, Y\})$, then $\mathcal{G}^+$ contains the following pattern: $U \leftrightarrow X \leftrightarrow Y \leftrightarrow V$, with U and V not adjacent in $\mathcal{G}^+$, and paths $V .. \rightarrow X$ and $U .. \rightarrow Y$ that do not go against an arrowhead.

For example, in the augmented skeleton in Figure 5(b), D-sep link $X - Z$ occurs in pattern $S \leftrightarrow X \leftrightarrow Z \leftrightarrow T$.

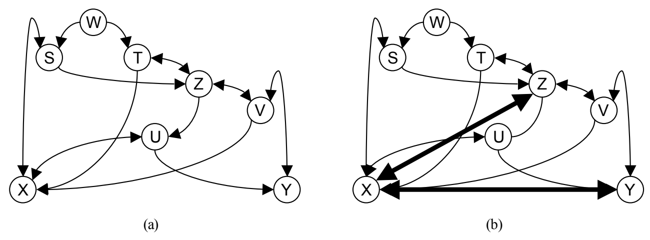


Figure 5: (a) Hierarchical D-sep links: $X \perp\!\!\!\perp Z \mid [S,T,W]$, with Z also appearing in $X \perp\!\!\!\perp Y \mid [S,T,U,V,W,Z]$; (b) corresponding augmented skeleton $\mathcal{G}^+$, with initially undiscovered D-sep links $X \leftrightarrow Z$ and $X \leftrightarrow Y$ (bold)

There is a hierarchical structure between D-sep links in the sense that if one pair of D-separated nodes is part of the D-separating set of another, then not the other way around.

Lemma 5. In a MAG $\mathcal{M}$, for two pairs of D-separable nodes $X \perp\!\!\!\perp Y \mid [\mathbf{Z}]$ and $X \perp\!\!\!\perp Y \mid [\mathbf{W}]$, if $X \in \mathbf{W}$ and/or $Y \in \mathbf{W}$, then $U \notin \mathbf{Z}$ and/or $V \notin \mathbf{Z}$.

It implies that we may need to find one D-sep link in a MAG $\mathcal{M}$ before we can find another, but there is no vicious circle of complex intertwined D-sep links.

Though all this will facilitate (and speed up) identification, it is not enough for a polynomial search algorithm, as this is dominated by the number of possible node-sets to consider for a single D-sep link.

4.2 Capturing the D-sep nodes

In this part we first show that all D-sep nodes needed to separate (X,Y) were already found as part of a minimal separating set between ancestors of $\{X,Y\}$. Then we show that we can recursively include these to obtain a D-separating set, starting from an appropriate set of nodes adjacent to X and Y.

Using shorthand $AA(\mathbf{X}) \equiv (Adj(\mathbf{X}) \cap An(\mathbf{X})) \setminus \mathbf{X}$ for the set of **adjacent ancestors** of $\mathbf{X}$, we find that:

Lemma 6. In a MAG $\mathcal{M}$, if $Z \in \mathbf{Z}$ is a D-sep node in $X \perp\!\!\!\perp Y \mid [\mathbf{Z}]$, then Z is also part of a minimal separating set between another pair of nodes from $\{X,Y\} \cup \mathbf{Z}_{\setminus Z} \cup AA(\{X,Y\})$, neither of which have selection bias.

As a result, if we know all minimal conditional independencies between nodes in $An(\{X,Y\})$, then all required D-separating nodes for $X \perp\!\!\!\perp Y \mid [\mathbf{Z}]$ already appear in one of these minimal separating sets. We want to use this information to guide the search for D-separating sets, but unfortunately we do not know what the ancestors are, and we do not have all minimal independencies (as the PC search only finds one minimal separating set for each eliminated edge).

Fortunately we can show that we do not need to know all minimal separating sets to find the relevant nodes, as it is sufficient to have just *one* minimal separating set for each nonadjacent pair, and recursively include these in the set. In formal terms:

Definition 4. An **independence set** $\mathcal{I} \subseteq \mathcal{I}(\mathcal{M})$ is a (sub)set of all minimal independence statements consistent with MAG $\mathcal{M}$, which contains at least one separating set for each pair of nonadjacent nodes in $\mathcal{M}$.

And a recursive definition for a set of separating nodes:

Definition 5. Let $\mathcal{I}$ be an independence set, then for a set $\mathbf{X}$ the **hierarchy** $HIE(\mathbf{X},\mathcal{I})$ is the union of $\mathbf{X}$ and all nodes that appear in a minimal separating set in $\mathcal{I}$ between any pair of nodes in $HIE(\mathbf{X},\mathcal{I})$.

By Lemma 2 all nodes in $HIE(\mathbf{X},\mathcal{I})$ are ancestors of one or more nodes in $\mathbf{X}$; see also Example 1, below.

With this the key result can be stated as:

Lemma 7. Let X and Y be D-separable nodes in a MAG $\mathcal{M}$. If independence set $\mathcal{I}$ contains at least one minimal separating set for each pair of nonadjacent nodes in $An(\{X,Y\})$ in $\mathcal{M}$ (except for $\{X,Y\}$ itself), then $HIE\big(AA(\{X,Y\}),\mathcal{I}\big)_{\setminus\{X,Y\}}$ is a D-separating set for X and Y.

In words: if X and Y are D-separable nodes, then they are separated by the hierarchy of minimal separating nodes implied by ancestors adjacent to X and/or Y.

If found, then the edge $X \leftrightarrow Y$ is removed from $\mathcal{G}^+$, and a corresponding minimal separating set is obtained by eliminating redundant nodes one by one until no more can be removed (Tian et al., 1998).

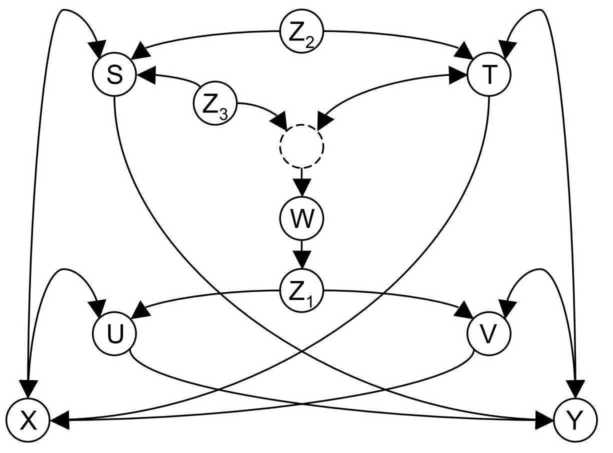


Figure 6: Hierarchical inclusion of separating sets ensures inclusion of D-sep node Z_3.

Example 1. In Figure 6, X and Y are D-separated by the set $\{S,T,U,V,Z_1,Z_2,Z_3\}$. Lemma 6 states that we can find D-sep node Z_3 from $S \perp\!\!\!\perp Z_1 \mid [Z_3]$, but the PC-stage may have found $S \perp\!\!\!\perp Z_1 \mid [W]$ instead, which would leave path $X \leftrightarrow S \leftrightarrow W \leftrightarrow T \leftrightarrow Y$ unblocked. However, Lemma 7 ensures that, whatever the independencies found by PC, node Z_3 is included in $HIE(\{X,Y,S,T,U,V\},\mathcal{I})$; indeed, we find Z_3 from either $S \perp\!\!\!\perp Z_1 \,|\, [Z_3]$ and/or $S \perp\!\!\!\perp W \,|\, [Z_3]$.

At this point, for a given suspect D-sep link $X \leftrightarrow Y$ in $\mathcal{G}^+$ we do not know exactly which nodes are in $AA(\{X,Y\})$ in $\mathcal{M}$. However, given that $\forall\{X,Y,Z\}$:

1. $AA(\{X,Y\}) \subseteq Adj(X) \cup Adj(Y)$,
2. $|Adj(X)| \leq k$ in $\mathcal{M}$, and
3. $Z \in Adj(X)$ in $\mathcal{M} \;\Rightarrow\; Z \in Adj(X)$ in $\mathcal{G}^+$,

we can be sure that the set $AA(\{X,Y\})$ in $\mathcal{M}$ consists of a combination of *at most* k nodes from $Adj(X)$ and k nodes from $Adj(Y)$ in $\mathcal{G}^+$.

For bounded k those can be searched in worst case polynomial time $N^k \times N^k = N^{2k}$, and so, if $X \leftrightarrow Y$ is indeed a D-sep link, we are guaranteed to find a separating set within that number of steps.

4.3 Search strategy for *D*-sep links

There is just one more aspect before we can turn the previous results into a complete causal discovery algorithm: Lemma 7 assumes that a minimal separating set is known for each separable pair in the ancestors of the (possible) D-sep link $X \leftrightarrow Y$. If these in turn also contain D-sep links, as for example $X \leftrightarrow Z$ in Figure 5(b), then we may need to find one before we can find the other. Therefore, if we find a new D-separating set we have to check previously tried possible D-sep edges in case the new set introduces more nodes into the corresponding hierarchies, after updating $\mathcal{G}^+$. By Lemma 5, as long as there are undiscovered D-sep links (still corresponding to edges in $\mathcal{G}^+$), then there is always at least one that cannot have undiscovered D-sep links between its ancestors, and so can be recognized as the pattern in Lemma 4 in the (updated) augmented skeleton $\mathcal{G}^+$. As a result, the procedure is guaranteed to terminate only after all are found. Revisiting possible D-sep links incurs an additional complexity factor N^2.

Finally, in the discussions so far we largely ignored the impact of possible selection bias, as all proofs remain valid with or without selection effects, see details in (Claassen et al., 2013). From (Richardson and Spirtes, 2002, §4.2.1) we know that selection on a node effectively destroys arrowheads on its ancestors, i.e. removes detectable non-ancestor relations. In particular, adding selection effects can overrule the D-sep pattern from Lemma 4, and either turn it into a regular separable pair (found by PC) or destroy the independence altogether. As a result, D-sep links are much rarer when selection bias is present, and the search for D-sep nodes can often be avoided altogether.

5 Implementing C-LEARN as FCI+

We incorporate the previous results into a modified **D-SEP** search of the FCI algorithm to obtain a sound and complete method for constraint-based causal discovery that is worst-case polynomial in the number of independence tests between N variables, provided the model is sparse (bounded by k). It assumes faithfulness and an underlying causal DAG, but does allow for latent variables and selection bias.

Algorithm 2 FCI+ algorithm

```
    In  : variables $\mathbf{V}$, oracle $\mathcal{O}$, sparsity $k$
    Out: causal model $\mathcal{G}$ over $\mathbf{V}$
 1: $\mathcal{G},\mathcal{I} \leftarrow PCAdjSearch(\mathbf{V},\mathcal{O},k)$
 2: $\mathcal{G}^+ \leftarrow AugmentGraph(\mathcal{G},\mathcal{I},\mathcal{O})$
 3: $PosDsepLinks \leftarrow GetPDseps(\mathcal{G}^+)$
    D-SEP search
 4: while $PosDsepLinks \neq \varnothing$ do
 5:     $X,Y \leftarrow Pop(PosDsepLinks)$
 6:     $BaseX \leftarrow Adj(X)_{\setminus Y}$
 7:     $BaseY \leftarrow Adj(Y)_{\setminus X}$
 8:     for $n = 1..k$ do
 9:         for $m = 1..k$ do
10:             get subset $\mathbf{Z}_X \subseteq BaseX$, size $n$
11:             get subset $\mathbf{Z}_Y \subseteq BaseY$, size $m$
12:             $\mathbf{Z}^* \leftarrow HIE(\{X,Y\} \cup \mathbf{Z}_X \cup \mathbf{Z}_Y,\mathcal{I})_{\setminus\{X,Y\}}$
13:             if $X \perp\!\!\!\perp Y \,|\, \mathbf{Z}^*$ then
14:                 $\mathbf{Z} \leftarrow MinimalDsep(X,Y,\mathbf{Z}^*)$
15:                 $\mathcal{I} \leftarrow UpdateSepsets(\mathcal{I},X,Y,\mathbf{Z})$
16:                 $\mathcal{G}^+ \leftarrow AugmentGraph(\mathcal{G}^+,\mathcal{I},\mathcal{O})$
17:                 $PosDsepLinks \leftarrow GetPDseps(\mathcal{G}^+)$
18:                 (continue while)
19:             end if
20:         end for
21:     end for
22: end while
23: $\mathcal{G} \leftarrow RunOrientationRulesFCI(\mathcal{G},\mathcal{I})$
24: return causal model $\mathcal{G}$
```

The FCI+ algorithm in Algorithm 2 starts in line 1 from the output of the PC adjacency search (line 16 in Algorithm 1), that is the skeleton $\mathcal{G}$ and minimal *Sepset* in $\mathcal{I}$ for each eliminated edge. Line 2 constructs the subsequent *augmented skeleton* $\mathcal{G}^+$ by testing for single node additions that destroy the independence, which is the basis for identifying the edges corresponding to possible D-sep links in line 3. This list is processed (and updated along the way) until no more unchecked possible D-sep links remain. For a pair of nodes $X \leftrightarrow Y$ on a possible D-sep edge in $\mathcal{G}^+$ the '*Base*' of adjacent nodes (possible ancestors) is determined in lines 6/7. For each combination of max. k nodes from this base around X and max. k nodes from the base around Y the corresponding hierarchy is computed, and tested for independence in line 13. If found it is turned into a minimal separating set, and stored in the list of sepsets $\mathcal{I}$. This is used to update the augmented skeleton $\mathcal{G}^+$ (remove edge and check for single node dependencies), and to update the set of possible D-sep links, e.g. in case we have to reconsider previously rejected edges or can now eliminate other candidates. Finally line 23 runs the standard FCI-orientation rules on the skeleton to return the target causal model in the form of a complete PAG.

Complexity analysis

Now to derive a bound on the worst-case complexity of the FCI+ algorithm. As stated, we assume a causal model in the form of a completed PAG $\mathcal{G}$ over N (observed) variables, with node degree $\leq k$, and a constant-time independence oracle.

Contributions of various stages in Algorithm 2:

- l.1: *PCAdjSearch* is order $O(N^{k+2})$, as it searches for subsets $\leq k$ nodes from $N-2$ variables to separate N^2 nodes on edges (Spirtes et al., 2000)
- l.2: *AugmentGraph* is order $O(N^3)$, given tests for at most $N-2$ nodes for $1/2N^2$ eliminated edges,
- l.3: *GetPDseps* could find up to $O(N^2)$ possible edges to process,
- l.10-12: there are $O(N^{2k})$ different (implied) combinations for possible D-sep sets in the hierarchy from two subsets of at most k nodes from $N-2$ variables,
- l.14: *MinimalDsep* is at most order $O(N^2)$, as nodes can be removed one-by-one (Tian et al., 1998)
- l.16: *AugmentGraph* is again order $O(N^3)$,
- l.17: *GetPDseps* might put back previously tried-but-failed D-sep edges, leading to a factor $O(N^2)$,
- l.23: *RunOrientationRulesFCI* is order $O(N^4)$: for N^2 edge marks it checks for certain paths in $\mathcal{G}$ which can be done in $O(N^2k)$ by a 'reachability' algorithm, with $Nk \sim$ nr. of edges, (Zhang, 2008, p.1881)

The dominant terms are $O(N^{2k})$ possible hierarchies to test for each of $O(N^2)$ possible edges which may have to be repeated $O(N^2)$ times, leading to an overal worst-case complexity of order $O(N^{2(k+2)})$. In other words: worst-case $O(FCI+) \sim O(PC^2)$, but both with a running time that is polynomial in the number of variables N given maximum node degree k.

Clearly, these bounds can be tightened, and many steps can be implemented more efficiently. For example the *Augmentgraph* procedure can be realized using rules on the available $\mathcal{I}$ (instead of additional independence tests), and include invariant tails as well. The number of $BaseX/Y$ combinations to test can be reduced by eliminating adjacent nodes that cannot be ancestor of X or Y, and making sure to always include necessary nodes. The hierarchy can be pre-computed for each pair of nodes, and only needs a (partial) update when a new D-sep link is found, etc. However, we are not (yet) looking for an optimal implementation, just to verify that a polynomial solution is possible.

6 Discussion

We have shown that it is possible to learn a sound and complete sparse causal model representation that is polynomial in the number of independence tests, even when latent variables and selection bias may be present. We presented an implementation in the from of the FCI+ algorihtm, which derives from standard FCI but with a modified *D-SEP* search stage.

The exponential complexity can be avoided by exploiting the inherent structure in the problem that stems from the underlying causal network. The problem can be broken down into a series of smaller steps, in contrast to, e.g. finding a minimal Bayesian network or a travelling salesman problem, where the minimal solution only applies to the specific network as a whole. This 'breaking down into subproblems' follows naturally in the constraint-based paradigm; an intriguing question is whether it is also possible to find a polynomial score-based causal discovery method.

Effectively, FCI+ uses independence information from previous stages to guide the *D-SEP* search, leading to a theoretical worst-case running time $O(N^{2(k+2)})$ in the number of independence tests. This is significantly better than exponential, but still unfeasible for large N. However, in practice the typical performance is vastly better: very few candidate D-sep links with even fewer repeats, and in particular much smaller sizes for the adjacent set $An(\{X,Y\}) \sim O(2k)$ for $BaseX/Y$ (line 6/7 in Algorithm 2). This leads to a dramatic reduction in actual runtime, as the most expensive term reduces from $O(N^{2k})$ to $O(2^{2k})$ base combinations for the D-sep set candidates.

A drawback for sample versions of FCI+ is that the conditioning sets may become larger than strictly necessary, leading to a loss of statistical power of the independence tests. Preliminary investigations in random graphs up to 200 variables suggest this effect is not very prominent (often the largest set remains smaller than for standard FCI), but it may become an issue in certain circumstances. An interesting solution is to limit the maximum node sets by taking the *intersection* of the hierarchical **D-SEP** candidate and FCI's **Possible-D-SEP** sets. Furthermore, many tests in the AugmentGraph procedures (line 2,16) may be avoided by applying some of FCI's orientation rules to the available strucuture. Finally we can try to restrict the number of possible D-sep-edges to check even further, and/or optimize the sequence in which to process them. Ultimately our goal is to come as close to the PC runtime as possible, while still handling hidden variables correctly.

Acknowledgements

Tom Claassen and Tom Heskes were supported by NWO Vici grant nr.639.023.604; Joris Mooij was supported by NWO Veni grant 639.031.036.

Appendix A. Proofs

This section contains proof sketches for the results in this paper; for details and examples, see (Claassen et al., 2013).

Lemma 3. *(D-separated nodes are non-ancestors)*

Proof sketch. In the supplement we show that if X is not independent of Y for any subset $Adj(X)$, then Y is not an ancestor of X and has no selection bias. For a D-sep link (X, Y) this applies to both; statement for minimal $\mathbf{Z}$ follows immediately from Lemma 2-(2). □

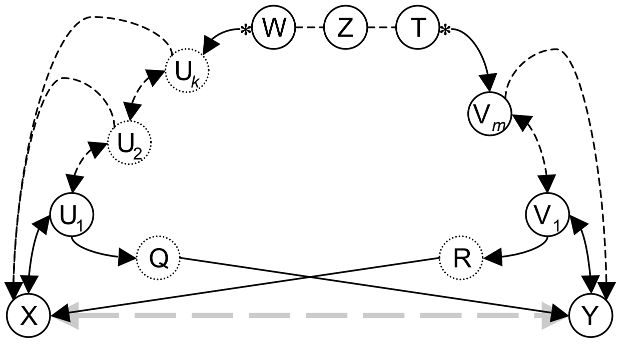


Figure 7: Path configuration for D-sep link $X \cdots Y$.

Lemma 4. *(bi-directed D-sep patterns in $\mathcal{G}^+$)*

Proof sketch. In the supplement we show that the path configuration in Figure 7 is always present for a D-sep edge $X - Y$, resulting in identifiable independencies $X \perp\!\!\!\perp W \,|\, [\mathbf{Z}']$, $Y \perp\!\!\!\perp T \,|\, [\mathbf{Z}'']$, and $U_1 \perp\!\!\!\perp V_1 \,|\, [\mathbf{Z}''']$ (for some sets $\mathbf{Z}'$ etc.). Furthermore, the first necessarily becomes dependent when including either U_1 or Y into the separating set. Idem for the second when including V_1 or X, as for the third when including X or Y. In combination with Lemma 2-(1) this implies identifiable invariant arrowheads in $\mathcal{G}^+$ on all three edges $U_1 \leftrightarrow X \leftrightarrow Y \leftrightarrow V_1$, with U_1 not adjacent to V_1. □

Lemma 5. *(D-sep link hierarchy)*

Proof sketch. Follows from Lemma 3, otherwise it would introduce a cycle. □

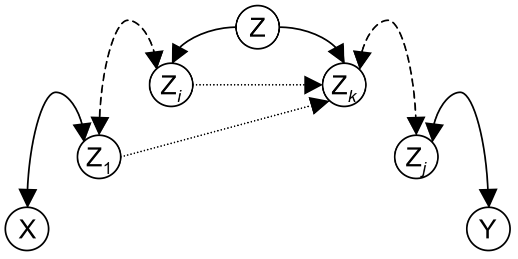


Figure 8: Canonical path blocked by D-sep node Z.

Lemma 6. *(all required D-sep nodes appear in other minimal separating sets we already found)*

Proof sketch. In the supplement we show that each D-sep node Z blocks a path of the generic form depicted in Figure 8. From this we construct a necessary non-adjacency of neighboring node Z_k with at least one of the other $\{Z_1, .., Z_i\}$ or X along the path, in which Z is part of a minimal set that separates them. □

Lemma 7. *(for D-sep link $X \leftrightarrow Y$ in $\mathcal{G}^+$ and independence set $\mathcal{I}$, the set $HIE(AA(\{X, Y\}), \mathcal{I})_{\setminus\{X,Y\}}$ is a D-separating set)*

Proof sketch. In the supplement we show that for a D-sep link the set of nodes in the hierarchy $HIE(\{X, Y\}, \mathcal{I})$ is independent of the specific (minimal) independence found for each pair of nodes that are not adjacent in $\mathcal{G}^+$. We do this by showing that if a node W is *optional* in a minimal separating set between two nodes X and Y, so both $X \perp\!\!\!\perp Y \,|\, [\mathbf{Z}]$ and $X \perp\!\!\!\perp Y \,|\, [\mathbf{W}]$ exist, with $W \in (\mathbf{W} \setminus \mathbf{Z})$, then there is also an optional node $Z \in (\mathbf{Z} \setminus \mathbf{W})$, and W is part of a minimal separating set between Z and either X or Y. This argument can be repeated until the node is necessary in some minimal separating set, and will be found in all independence sets $\mathcal{I}$. So every node that is optional in one minimal separating set (and so may not be included in the corresponding Sepset from the PC adjacency search) is a necessary node in a Sepset between some other pair when 'zooming in' on the graph. As these are all included, it follows that $HIE(\{X, Y\}, \mathcal{I})$ is independent of the specific separating sets found in $\mathcal{I}$. By Lemma 6, all D-sep nodes we need to separate X and Y als appear in some minimal set between two nodes from the union of $\{X, Y\}$ and the corresponding D-separating set, for which the same applies again, until ultimately an independence between two nodes in $Adj(\{X, Y\}) \cup \{X, Y\}$ is reached. But that means they are all part of $HIE(AA(\{X, Y\}), \mathcal{I})_{\setminus\{X,Y\}}$, and so this is guaranteed to be a D-separating set for $X \leftrightarrow Y$. □

References

D. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3(3):507–554, 2002.

D. Chickering, D. Heckerman, and C. Meek. Large-sample learning of Bayesian networks is NP-hard. *J. Mach. Learn. Res.*, 5:1287–1330, December 2004.

T. Claassen and T. Heskes. A logical characterization of constraint-based causal discovery. In *Proc. of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 135–144, 2011.

T. Claassen, J. Mooij, and T. Heskes. Proof supplement to Learning sparse causal models is not NP-hard. Technical report, Faculty of Science, Radboud University Nijmegen, 2013. `http://www.cs.ru.nl/~tomc/docs/NPHardSup.pdf`.

D. Colombo, M. Maathuis, M. Kalisch, and T. Richardson. Learning high-dimensional DAGs with latent and selection variables. *The Annals of Statistics*, 40(1):294–321, 2012.

G. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, (42):393–405, 1990.

J. Cussens. Bayesian network learning with cutting planes. In *Proc. of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 153–160. AUAI Press, 2011.

R. Evans and T. Richardson. Maximum likelihood fitting of acyclic directed mixed graphs to binary data. In *Proc. of the 26th Conference on Uncertainty in Artificial Intelligence*, pages 177–184, 2010.

M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Co., 1979.

O. Goldreich. *Computational Complexity: A Conceptual Perspective.* Cambridge University Press, 2008.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* The MIT Press, 2009.

D. Margaritis and S. Thrun. Bayesian network induction via local neighborhoods. In *Advances in Neural Information Processing Systems 12*, pages 505–511, 1999.

J. Pearl. *Causality: models, reasoning and inference.* Cambridge University Press, 2000.

J. Pearl and T. Verma. A theory of inferred causation. In *Knowledge Representation and Reasoning: Proc. of the Second Int. Conf.*, pages 441–452, 1991.

T. Richardson and P. Spirtes. Ancestral graph Markov models. *Ann. Stat.*, 30(4):962–1030, 2002.

P. Spirtes, C. Meek, and T. Richardson. An algorithm for causal inference in the presence of latent variables and selection bias. In *Computation, Causation, and Discovery*, pages 211–252. AAAI Press, Menlo Park, CA, 1999.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search.* The MIT Press, Cambridge, Massachusetts, 2nd edition, 2000.

J. Tian, A. Paz, and J. Pearl. Finding minimal d-separators. Technical Report R-254, UCLA Cognitive Systems Laboratory, 1998.

C. Yuan and B. Malone. An improved admissible heuristic for learning optimal Bayesian networks. In *Proc. of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 924–933, Corvallis, Oregon, 2012. AUAI Press.

J. Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16-17):1873 – 1896, 2008.

Advances in Bayesian Network Learning using Integer Programming

Mark Barlett
Dept of Computer Science &
York Centre for Complex Systems Analysis
University of York, UK
`mark.bartlett@york.ac.uk`

James Cussens
Dept of Computer Science &
York Centre for Complex Systems Analysis
University of York, UK
`james.cussens@york.ac.uk`

Abstract

We consider the problem of learning Bayesian networks (BNs) from complete discrete data. This problem of discrete optimisation is formulated as an integer program (IP). We describe the various steps we have taken to allow efficient solving of this IP. These are (i) efficient search for cutting planes, (ii) a fast greedy algorithm to find high-scoring (perhaps not optimal) BNs and (iii) tightening the linear relaxation of the IP. After relating this BN learning problem to set covering and the multidimensional 0-1 knapsack problem, we present our empirical results. These show improvements, sometimes dramatic, over earlier results.

1 Introduction

Bayesian networks (BNs) use a directed acyclic graph (DAG) to represent conditional independence relations between random variables. Each BN defines a joint probability distribution over joint instantiations of the random variables. BNs are a very popular and effective representation for probabilistic knowledge and there is thus considerable interest in 'learning' them from data (and perhaps also prior knowledge).

In this paper we present our approach to the 'search-and-score' method of BN learning. As a score for any candidate BN we choose the log marginal likelihood with Dirichlet priors over the parameters. We choose these Dirichlet priors so that our score is the well-known BDeu score. Throughout we set the *effective sample size* for the BDeu score to be 1. The learning problem is a discrete optimisation problem: find the BN structure (DAG) with the highest BDeu score.

In Section 2 we present our integer programming (IP) encoding of the problem. Section 3 shows how we use a 'branch-and-cut' approach to solve the IP problem. Section 4 describes how we find 'cutting planes'. Section 5 describes our fast greedy algorithm for finding good but typically sub-optimal BNs. Section 6 presents our investigation into the convex hull of DAGs with 3 or 4 nodes, the facets of which turn out to be useful for learning DAGs with any number of nodes. Section 7 is more discursive in nature: pointing out interesting connections to the set covering and multi-dimensional knapsack problems. The central contribution of the paper—faster solving for exact BN learning—is established in Section 8 where we present our empirical results. We finish, as is the custom, with conclusions and pointers to future work.

This paper assumes familiarity with Bayesian networks and basic knowledge concerning BN learning. See [14] for a comprehensive treatment of both these topics and much more besides. The rest of the paper assumes the reader knows what an integer program is: it is a discrete optimisation problem where the objective function is a linear function of the (discrete) problem variables and where the set of feasible solutions is defined by a finite set of linear inequalities over these variables. The *linear relaxation* of the IP is obtained by removing the constraint that the problem variables take only integer values. For further reading on integer programming, consult e.g. [16]. We refer to the variables in the learned BN structure as 'nodes' rather than 'variables' to more clearly distinguish them from IP variables.

2 Integer program encoding

We encode our BN structure learning problems as an integer program (IP) so that the log marginal likelihood of any DAG is a linear function of the IP variables. This requires creating binary 'family' variables $I(W \to v)$ for each node v and candidate parent set W, where $I(W \to v) = 1$ iff W is the parent set for v. This encoding has been used in previous work [5, 13, 6].

The objective function then becomes

$$\sum_{v,W} c(v,W)I(W \to v) \tag{1}$$

where $c(v,W)$ is the 'local' BDeu score for W being the parents of v. This score is computed from the data prior to the search for an optimal BN. Evidently the number of such scores/variables increases exponentially with p, the number of nodes in the BN. However if $W \subset W'$ and $c(v,W) > c(v,W')$ then, in the absence of any user constraints on the DAG, it is obvious that W cannot be the parent set for v in any optimal BN and thus there is no need to create the unnecessary variable $I(W' \to v)$. Moreover, using a pruning technique devised by de Campos and Ji [9] it is possible to prune the search for necessary variables so that local scores for many unnecessary variables need never be computed. Nonetheless, unless p is small (e.g. ≤ 12), even with this pruning we typically have to make some restriction on the number of candidate parent sets for any node. See Section 8 for more on this.

The $I(W \to v)$ variables encode DAGs, but in many cases one is more interested in *Markov equivalence classes* of DAGs [14] which group together DAGs representing the same conditional independence relations. This has motivated the *standard imset* and later *characteristic imset* representation [12]. A characteristic imset c is a zero-one vector indexed by subsets C of BN nodes such that $\mathsf{c}(C) = 1$ iff there is a node $v \in C$ such that all nodes in $C \setminus \{v\}$ are parents of v. Importantly two DAGs have the same characteristic imset representation iff they are Markov equivalent. This representation lends itself to an IP approach to BN structure learning. Existing [12] and ongoing work shows encouraging results. Connecting the characteristic imset representation to our 'family' representation we have that for any DAG and any subset of nodes C:

$$\mathsf{c}(C) = \sum_{v \in C} \sum_{W: C \setminus \{v\} \subseteq W} I(W \to v) \tag{2}$$

Returning to our own encoding, to ensure that IP solutions correspond to DAGs two families of linear constraints are used. Let V be the set of BN nodes. The convexity constraints

$$\forall v \in V : \sum_{W} I(W \to v) = 1 \tag{3}$$

simply ensure that any variable has exactly one parent set. The 'cluster' constraints

$$\forall C \subseteq V : \sum_{v \in C} \sum_{W: W \cap C = \emptyset} I(W \to v) \geq 1 \tag{4}$$

introduced by Jaakkola *et al* [13], ensure that the graph has no cycles. For each cluster C the associated constraint declares that at least one $v \in C$ has no parents in C. Since there are exponentially many cluster constraints these are added as 'cutting planes' in the course of solving.

3 Branch-and-cut algorithm

Let n be the number of $I(W \to v)$ variables so that any instantiation of these variables can be viewed as a point in $\{0,1\}^n \subset [0,1]^n \subset \mathbb{R}^n$. Let $\mathcal{P}_{cluster}$ be the polytope of all points in $\mathbb{R}^n$ which satisfy the bounds on the variables, convexity constraints (3) and cluster constraints (4). If a point in $\mathcal{P}_{cluster}$ is entirely integer-valued (i.e. is in $\{0,1\}^n$) then it corresponds to a DAG, but $\mathcal{P}_{cluster}$ also contains infinitely many non-DAG points. Due to the p convexity equations (3) this polytope is of dimension $n - p$. Note that all points in $\mathcal{P}_{cluster}$ have an objective value (1) not just those with integer values.

Jaakkola *et al* [13] consider the problem of finding a point in $\mathcal{P}_{cluster}$ with maximal objective value. They consider the dual problem and iteratively add dual variables corresponding to cluster constraints. This process is continued until the problem is solved or there are too many dual variables. In the latter case branch-and-bound is used to continue the solving process. A 'decoding' approach is used to extract DAGs from values of the dual variables.

We instead take a fairly standard 'branch-and-cut' approach to the problem. The essentials of branch-and-cut are as follows, where *LP solution* is short for 'solution of the current linear relaxation':

```
1. Let x* be the LP solution.
2. If there are valid inequalities
       not satisfied by x*
       add them and go to 1.
   Else if x* is integer-valued then
       the current problem is solved
   Else branch on a variable with
       non-integer value in x*
       to create two new sub-IPs.
```

The added valid inequalities are called 'cutting planes' since they 'cut off' the LP solution x^*. This process of cutting and perhaps branching is performed on all nodes in the search tree. If the objective value of the LP solution of a sub-IP is worse than the current best integer solution then the tree is pruned at that point. Note that the term 'branch-and-cut' is a little misleading since there is typically much cutting in the root node before any branching is done.

Where Jaakkola *et al* added dual variables to the dual problem we add cluster constraints as cutting planes (*cluster cuts*) to the original 'primal' problem. We do a complete search for cluster cuts so that if none can be found we have a guarantee that the LP solution x^* satisfies all cluster constraints (4). Note that there is no need to find all cluster cuts to have this guarantee; in practice only a small fraction of the exponentially many cluster cuts need be found.

If no cluster cuts can be found (and the problem is not solved) we search for three other sorts of cutting planes: Gomory, strong Chvátal-Gomory (CG) and zero-half cuts. If none of these cuts can be found we branch on a fractional $I(W \to v)$ variable to create two new sub-IPs as described above. Solving the problem returns a guaranteed optimal BN. Since we are working with the primal representation there is no decoding required to extract the BN from the optimal values of the $I(W \to v)$ variables: we just return the p variables which are set to 1.

The algorithm is implemented using the the SCIP [1] callable library (`scip.zib.de`). Implementing a basic branch-and-cut algorithm with SCIP amounts to little more than setting SCIP parameter flags appropriately. We used SCIP's built-in functions to search for Gomory, strong CG and zero-half cuts. We also used SCIP's default approach to branching. See [1] and the SCIP documentation for details. However the search for cluster cuts we implemented ourselves, plugging it into the branch-and-cut algorithm as a SCIP *constraint handler*. This search is described in Section 4. Our greedy algorithm for finding good BNs is implemented as a SCIP *primal heuristic* and is described in Section 5.

4 Finding cluster cuts

Our system GOBNILP 1.3 uses a sub-IP to search for cluster cuts. The approach is essentially the same as that presented by Cussens [6]. Nonetheless we describe it here for completeness and because our current implementation has a simple but very effective optimisation that was missing from the earlier one.

To understand the sub-IP first note that, due to the convexity constraints, the constraint (4) for cluster C can be reformulated as a knapsack constraint (5):

$$\sum_{v \in C} \sum_{W : W \cap C \neq \emptyset} I(W \to v) \leq |C| - 1 \tag{5}$$

In the sub-IP the cluster is represented by $|V| = p$ binary variables $I(v \in C)$ each with objective coefficient of -1. Now let $x^*(W \to v)$ be the value of $I(W \to v)$ in the LP solution. For each $x^*(W \to v) > 0$ create a sub-IP binary variable $J(W \to v)$ with an objective coefficient of $x^*(W \to v)$. Abbreviating $\sum_{v \in V} I(v \in C)$ to $|C|$ the sub-IP is defined to be:

$$\text{Maximise: } -|C| + \sum x^*(W \to v) J(W \to v) \tag{6}$$

Subject to, for each $J(W \to v)$:

$$J(W \to v) \Rightarrow I(v \in C) \tag{7}$$

$$J(W \to v) \Rightarrow \bigvee_{w \in W} I(w \in C) \tag{8}$$

The constraints (7) and (8) are posted as SCIP `logicor` clausal constraints and the sub-IP search is set to depth-first, branching only on $I(v \in C)$ variables. Note that `logicor` constraints are a special type of linear constraint. It is a requirement that the objective value is greater than -1. For efficiency this is implemented directly as a limit on the objective value (using SCIP's `SCIPsetObjlimit` function) rather than as a linear constraint. A final constraint dictates that $|C| \geq 2$.

In any feasible solution we have that

$$-|C| + \sum x^*(W \to v) J(W \to v) > -1 \tag{9}$$

Due to the constraints (7) and (8), for $J(W \to v)$ to be non-zero v must be in the cluster C and at least one element of W must also be in C. So (9) can be rewritten as:

$$-|C| + \sum_{v \in C} \sum_{W : W \cap C \neq \emptyset} x^*(W \to v) J(W \to v) > -1 \tag{10}$$

It follows that the cluster C associated with a feasible solution of the sub-IP has a cluster constraint which is violated by the current LP solution x^*. Each feasible solution of the IP thus corresponds to a valid cutting plane. The sub-IP is always solved to optimality, (collecting any sub-optimal feasible solutions along the way) so it follows that if the current LP solution violates any cluster constraint then at least one will be found.

In [6] a sub-IP was also used to find cutting planes. However, there "a binary variable $J(W \to v)$ is created for **each** family variable $I(W \to v)$ in the main IP." (our emphasis). In the current approach $J(W \to v)$ variables are created *only for $I(W \to v)$ variables which are not zero in the LP solution.* This greatly reduces the number of variables in the sub-IP (and thus speeds up sub-IP solving) since typically most main IP variables *are* set to zero in any LP solution.

5 Sink finding algorithm

Finding a good primal solution (i.e. a BN) early on in the search is worthwhile even if it turns out to be

$I(W_{1,1} \to 1)$	$I(W_{1,2} \to 1)$	...	$I(W_{1,k_1} \to 1)$
$I(W_{2,1} \to 2)$	$I(W_{2,2} \to 2)$	...	$I(W_{2,k_2} \to 2)$
$I(W_{3,1} \to 3)$	$I(W_{3,2} \to 3)$	...	$I(W_{3,k_3} \to 3)$
...	...	...	...
$I(W_{p,1} \to p)$	$I(W_{p,2} \to p)$	...	$I(W_{p,k_p} \to p)$

Table 1: Example initial state of the sink-finding heuristic for $|V| = p$. Rows need not be of the same length.

~~$I(W_{1,1} \to 1)$~~	$I(W_{1,2} \to 1)$	...	$I(W_{1,k_1} \to 1)$
$\mathbf{I(W_{2,1} \to 2)}$	~~$I(W_{2,2} \to 2)$~~	~~...~~	~~$I(W_{2,k_2} \to 2)$~~
$I(W_{3,1} \to 3)$	~~$I(W_{3,2} \to 3)$~~	...	$I(W_{3,k_3} \to 3)$
...	...	...	...
~~$I(W_{p,1} \to p)$~~	~~$I(W_{p,2} \to p)$~~	...	$I(W_{p,k_p} \to p)$

Table 2: Example intermediate state of the sink-finding heuristic.

suboptimal. High scoring solutions allow more effective branch-and-bound and may help in the root node due to *root reduced cost strengthening* [1, §7.7]. If a problem cannot be solved to optimality having a good, albeit probably suboptimal, solution is even more important.

We have a 'sink-finding' algorithm which proposes primal solutions using the current LP solution. The algorithm is based on two ideas: (i) that there might be good primal solutions 'near' the current LP solution and (ii) that an optimal BN is easily found if we can correctly guess an optimal total ordering of BN nodes. The first idea is common to all *rounding heuristics*. SCIP has 6 built-in rounding heuristics and we allow SCIP to run those that are fast and they do sometimes find high scoring BNs. The second idea has been exploited in dynamic programming approaches to exact BN learning [15].

To understand how the algorithm works consider Table 1. The table has a row for each node and there are $p = |V|$ rows. The $I(W \to v)$ variables for each node are ordered according to their objective coefficient $c(v, W)$, so that, for example, $W_{1,1}$ is the 'best' parent set for node 1 and W_{1,k_1} the worst. The objective coefficients $c(v, W)$ play no role in the sink-finding algorithm other than to determine this ordering. Since the number of available parent sets may differ between nodes the rows will typically not be of the same length.

On the first iteration of the sink finding algorithm, for each child variable the 'cost' of selecting its best-scoring parent set is computed. This cost is $1 - x^*(W_{v,1} \to v)$, where $x^*(W_{v,1} \to v)$ is the value of $I(W_{v,1} \to v)$ in the LP solution.

Denote the child variable chosen as v_p. In order to ensure that the algorithm generates an acyclic graph a total order is also generated. This is achieved by setting $I(W \to v)$ to 0 if $v_p \in W$. As a result v_p will be a sink node of the BN which the algorithm will eventually construct.

Suppose that it turned out that $v_p = 2$ and that node 2 was a member of parent sets $W_{1,1}$, $W_{3,2}$, $W_{p,1}$ and $W_{p,2}$. The state of the algorithm at this point is illustrated in Table 2. In the next iteration $I(W_{3,1} \to 3)$ remains available as the 'best' parent set for node 3 but for node 1 the best parent set now is $I(W_{1,2} \to 1)$. In the second and subsequent iterations the algorithm continues to choose the best available parent set for some node according to which choice of node has minimal cost. However in these non-initial iterations cost is computed as $(\sum_{W \in \mathrm{ok}(v)} x^*(W \to v)) - x^*(W_{v,\mathrm{best}} \to v)$, where $I(W_{v,\mathrm{best}} \to v)$ is the best scoring remaining choice for v and $\mathrm{ok}(v)$ is the set of remaining parent set choices for v. After each such selection, parent set choices for remaining nodes are updated just like for the first iteration. Note that the node selected at any iteration will be a sink node (in the final DAG) for the subset of nodes available at that point.

There is an added complication if some $I(W \to v)$ are already fixed to 1 when the sink-finding algorithm is run. This can happen either due to user constraints or due to branching on $I(W \to v)$. Trying to rule out such a variable leads the algorithm to abort.

Due to its greedy nature the sink finding algorithm is very fast and so we can afford to run it after solving *every* LP relaxation. For example, in one of our bigger examples, `Diabetes_100`, the sink-finding algorithm was called 9425 times taking only 30s in total. Note that each new batch of cutting planes produces a new LP and thus a new LP solution. In this way we use the LP to help us move around to search for high-scoring BNs.

6 Tightening the LP relaxation

Given a collection of n $I(W \to v)$ variables, each feasible DAG corresponds to a (binary) vector in $\mathbb{R}^n$. Consider now $\mathcal{P}$, the *convex hull* of these points and recall $\mathcal{P}_{cluster}$, the polytope defined in Section 3 containing all points satisfying the variable bounds, the convexity constraints (3) and all cluster constraints (4). As Jaakkola *et al* [13] note, $\mathcal{P}$ is strictly contained within $\mathcal{P}_{cluster}$, except in certain special cases. GOBNILP uses SCIP to add Gomory, strong CG and zero-half cutting planes in addition to cluster cuts. This produces a linear relaxation which is tighter than $\mathcal{P}_{cluster}$ and, as the results presented in Section 8 show, typically improves overall performance, although strong CG cuts are generally not helpful. Denote the polytope defined by adding these 'extra' cuts by $\mathcal{P}'_{cluster}$. Since the searches for Gomory, strong CG and zero-half cuts are not complete $\mathcal{P}'_{cluster}$ is specific to the problem instance and SCIP parameter settings.

In many cases it is not possible to separate a fractional LP solution x^* even with these extra cuts, so we have $x^* \notin \mathcal{P}$ but $x^* \in \mathcal{P}'_{cluster}$. This raises the question of which inequalities are needed to define $\mathcal{P}$. We have approached this issue by carrying out empirical investigations into $\mathcal{P}$ when there are 3 or 4 nodes.

For 3 nodes $\{1,2,3\}$ there are only 25 DAGs. We eliminated the three $I(\emptyset \to v)$ variables using the equations (3) and encoded each DAG using the remaining nine $I(W \to v)$ variables (3 remaining choices of parent set for each node). The lrs algorithm [2] (`http://cgm.cs.mcgill.ca/~avis/C/lrs.html`) was used to find the facets of the convex hull of these 25 vertices in $\mathbb{R}^9$.

Denoting this convex hull as $\mathcal{P}_3$, we find that it has 17 facets. These are: 9 lower bounds on the variables, 3 inequalities corresponding to the original convexity constraints, cluster constraints for the 4 clusters $\{1,2\}$, $\{1,3\}$, $\{2,3\}$, and $\{1,2,3\}$ and one additional constraint:

$$I(\{2,3\} \to 1) + I(\{1,3\} \to 2) + I(\{1,2\} \to 3) \leq 1 \tag{11}$$

Consider the point y^* in $\mathbb{R}^9$ specified by setting $I(\{2,3\} \to 1) = \frac{1}{2}$, $I(\{1,3\} \to 2) = \frac{1}{2}$, $I(\{1,2\} \to 3) = \frac{1}{2}$ and all other variables to zero. It is not difficult to see that this point is on the surface of $\mathcal{P}_{cluster}$ (lying on the hyperplanes defined by the cluster constraints for $\{1,2\}$, $\{1,3\}$ and $\{2,3\}$). However it does not satisfy (11) and so is outside of $\mathcal{P}_3$. Note that there are nine 3-node DAGs where one node has two parents. All of these DAGs lie on the hyperplane defined by (11). It is easy to show that these 9 DAGs (= points in $\mathbb{R}^9$) are affinely independent which establishes that (11) is indeed a facet.

The point y^* was also discussed by Jaakkola *et al* [13]. They considered the *acyclic subgraph polytope* $\mathcal{P}_{dag}$ which is the convex hull of DAGs which results from using binary variables to represent *edges* rather than parent set choices. This polytope has been extensively studied in discrete mathematics [11] and many (but not all) classes of facets are known for it [10]. The parent set representation can be projected onto the edge representation (so that the former is an *extended formulation* of the latter in the language of mathematical programming). As Jaakkola *et al* observe the projection of y^* is a member of $\mathcal{P}_{dag}$.

The inequality (11) can be generalised to give a class of valid *set packing* inequalities:

$$\forall C \subseteq V : \sum_{v \in C} \sum_{W : C \setminus \{v\} \subseteq W} I(W \to v) \leq 1 \tag{12}$$

We have found that adding all non-trivial inequalities of this sort for $|C| \leq 4$ speeds up solving considerably (see Section 8). This is because the LP relaxation is tighter. Since there are not too many such inequalities they are added directly to the IP rather than being added as cutting planes. Note that making the connection to characteristic imsets with (2) implies these set packing constraints.

We have not found all facets of $\mathcal{P}_4$, which is a polytope in $\mathbb{R}^{28}$ with 543 vertices (for the 543 4-node DAGs). We terminated lrs after a week's computation, by which time it had found 64 facets. We detected 10 different types of facets which we have labelled 4A to 4J. We will provide a full description of these facet classes in a forthcoming technical report. Here we just give a brief overview.

Consider, as an example, 4B-type facets. They are specified as follows:

$$\begin{aligned} & \sum_{v_4 \in W \wedge \{v_2,v_3\} \cap W \neq \emptyset} I(W \to v_1) \\ & + \sum_{v_3 \in W \vee \{v_1,v_4\} \subseteq W} I(W \to v_2) \\ & + \sum_{v_2 \in W \vee \{v_1,v_4\} \subseteq W} I(W \to v_3) \\ & + \sum_{v_1 \in W \wedge \{v_2,v_3\} \cap W \neq \emptyset} I(W \to v_4) \quad \leq \quad 2 \end{aligned} \tag{13}$$

Consider the point $z^* \in \mathbb{R}^{28}$ where all variables take zero value except: $I(\{3,4\} \to 1) = \frac{1}{2}$, $I(\{1,3\} \to 2) = \frac{1}{2}$, $I(\{1,4\} \to 2) = \frac{1}{2}$, $I(\{2,4\} \to 3) = \frac{1}{2}$, $I(\{1,2\} \to 4) = \frac{1}{2}$. It is easy to check that z^* satisfies all cluster constraints and any constraint of type (12). However, setting $v_i = i$ in (13) we have that the left-hand side is $2\frac{1}{2}$ and so (13) separates (i.e. cuts off) z^*. It follows that adding 4B-type linear inequalities results in a strictly tighter linear relaxation.

We have implemented 6 distinct cutting plane algorithms to search for inequalities of types 4B, 4C, 4E, 4F, 4G and 4H. We call cuts of this sort *convex4* cuts. Type 4A cuts are cluster cuts, and cuts of type 4D, 4I and 4J have not appeared useful in preliminary experiments. We have also experimented with adding a limited number of convex4 cuts directly to the IP rather than finding them 'on the fly' as cutting planes.

In practice we have found LP solutions which violate these constraints but which none of our other cutting planes can separate (the example z^* was extracted from one such LP solution). Cuts of type 4B appear to be particularly useful. Preliminary experiments indicate that using convex4 cuts is typically but not always beneficial. We have yet to do a controlled evaluation, but using convex4 cuts with a different version of SCIP (SCIP 3.0) and a slightly different machine from that used to present our main results, we do have some partial preliminary results. Using convex4 cuts, problem instances `alarm_3_10000`, `carpo_3_100`, `carpo_3_1000`, `carpo_3_10000` and `Diabetes_2_100` took 54s, 612s, 92s, 660s and 1393s to solve, respectively. All of these times are better than using our properly evaluated system GOBNILP 1.3 (see Section 8). The improvement is particular dramatic for `alarm_3_10000` which takes 298s using GOBNILP 1.3 and took 12872s using the system presented by Cussens [6]. On the other hand, problems `hailfinder_3_1` and `Pigs_2_1000` took 139s and 2809s respectively which is slower than GOBNILP 1.3.

7 Set covering and knapsack representations

If the convexity constraints (3) are all relaxed to set covering constraints: $\forall v : \sum_W I(W \to v) \geq 1$ then the BN learning problem becomes a pure set covering problem—albeit one with exponentially many constraints—since all the cluster constraints (4) are already set covering constraints. It is not difficult to show that any optimal solution to the set covering relaxation of our BN learning problem is also an optimal solution to the original problem. This opens up the possibility of applying known results concerning the set covering polytope to the BN learning problem. In particular, Balas and Ng [3] provide conditions for set covering inequalities to be facets and also show that for an inequality with integer coefficients and right-hand side of 1 to be a facet it must be one of the set covering inequalities defining the IP. This is a useful result for BN learning. It shows there is no point looking for 'extra' set covering inequalities in the hope they might be facets. Relaxed convexity constraints and cluster constraints are the only set covering inequalities that can be facets.

As Balas and Ng [3] note, Chvátal's procedure [4] can be used to generate the convex hull of integer points satisfying an IP after a finite number of applications. They provide a specialised version of this procedure to find a class of valid inequalities of the form $\alpha^S x \geq 2$ for the set covering polytope. These inequalities dominate all other valid inequalities of the form $\beta x \geq 2$. Each such inequality is defined by taking a set S of set covering inequalities, and combining them to get a new inequality (details omitted for space, see [3]).

We can use the procedure to get new inequalities for the BN learning problem by combining cluster constraints. For example combining the constraints for $C = \{a,b\}, C = \{a,c\}, C = \{a,d\}$ produces: $\sum_{W:W\cap\{b,c,d\}=\emptyset} 2I(W \to a) + \sum_{W:0<|W\cap\{b,c,d\}|<3} I(W \to a) + \sum_{a\notin W} I(W \to b) + \sum_{a\notin W} I(W \to c) + \sum_{a\notin W} I(W \to d) \geq 2$.

Our BN learning problem can also be reformulated as a multi-dimensional 0-1 knapsack problem. Due to the convexity constraints (4) we can eliminate each $I(\emptyset \to v)$ variable by replacing it with the linear expression $1 - \sum_{W\neq\emptyset} I(W \to v)$. Due to pre-pruning, the objective value of $I(\emptyset \to v)$ will be lower than that for all other $I(W \to v)$ variables and so once the $I(\emptyset \to v)$ variables have been eliminated the remaining variables will all have positive objective coefficient. Eliminating $I(\emptyset \to v)$ variables from the cluster constraints produces the knapsack constraints (5) previously mentioned.

8 Results

The system GOBNILP 1.3 described in the previous sections was implemented using C, with SCIP 2.1.1 used as the constraint solver. The underlying LP solver was CPLEX 12.5. Both SCIP and CPLEX are available for free under academic licences. All experiments were performed using a single core of a 64-bit Linux machine with a 2.80 GHz 4 core processor and 7.7 GB of RAM. A time out limit of 2 hours was imposed across all experiments after which runs were aborted. Our results can be reproduced by going to `http://www.cs.york.ac.uk/aig/sw/gobnilp`.

Experiments were performed on data taken from a variety of Bayesian networks, with different numbers of observations, N, and with different limits, m, on the maximum number of nodes considered as the parent set of each node. The problem sets used are shown in Table 3. Several of these problem sets were used in [6], with additional larger networks and parent set sizes being added to assess performance on harder problems. Local BDeu scores were computed for all experiments external to the systems tested and the times taken to compute and filter these are not included in the presented results. Score computation times ranged from 1 second to 5497 seconds in the longest case, `diabetes` with $N = 10000$.

The primary experiment in this paper is to assess how long GOBNILP 1.3 takes to find the BN with the highest score, and rule out the possibility of finding a BN with a higher score for each dataset. In particular, we compare how the system with all features introduced in previous sections compared to the earlier IP-based BN learning system presented by [6] (henceforth referred to as Cussens 2011).

Additionally, we examine the behaviour of the systems in those situations in which they failed to find the highest scoring BN within the 2 hour time limit. Integer Programming can be used as an any-time learning algorithm, where a current best solution can be taken at any point during the search, though this may not turn out to be the eventual best solution. Specifically, our aim here is to examine the examples which reached the two hour solving time limit and determine how close to finding a provably best BN they are at that point. This gives an idea of how good that system would be for use as an any-time algorithm, and acts as an (imperfect) proxy for comparing how much longer the search process would take to reach completion.

The Cussens 2011 system is not publicly available. However, GOBNILP 1.0 is available and is closely based on Cussens 2011 with some inefficiencies taken out. Therefore, comparisons were performed against GOBNILP 1.0 using exactly the same machine and SCIP and CPLEX versions as those used to run GOBNILP 1.3. These results are shown in Table 3. As the results reported in [6] are performed on a broadly similar machine to that used for the current experiments, times taken directly from that paper are also shown in the table for illustrative purposes. Results are not shown in the table for some data sets, as [6] did not study and report them.

The results show that in the vast majority of cases, GOBNILP 1.3 outperforms GOBNILP 1.0, often being 2–3 times faster. GOBNILP 1.3 is also never slower than Cussens 2011 and for the larger examples is usually an order of magnitude quicker. For example, the `alarm 3 10000` data set takes over 3.5 hours to solve using Cussens 2011, but less than 5 minutes using GOBNILP 1.3. Some of the difference in run times between GOBNILP 1.3 and Cussens 2011 may be due to different machines being used, however the vastly improved performance on the larger examples undoubtedly reflects an overwhelming improvement.

In order to discover which aspects of GOBNILP 1.3 were leading to this improvement in performance, a second set of experiments was conducted in which the performance of the full system was compared to that resulting from removing parts of the system one at a time. Three features were identified as being suitable to remove while still leaving a system that would still result in the best BN being found, albeit potentially not as efficiently. These three features were

Set Packing Constraints (Section 6) As these (12) are logically implied by the basic problem, without them the IP for finding the BN is still correct.

Sink Primal Heuristic (Section 5) The algorithm for finding feasible solutions through sink finding is not necessary, but may improve the search process through tightening the lower bound on the best BN.

Value Propagator Explicitly determining which values must be fixed at zero or one at each node of the search tree is not necessary as this information will eventually be discovered through search. However, by performing this propagation as early as possible, a significant amount of search may be saved.

In addition, three cutting plane algorithms which are built into SCIP are used within GOBNILP 1.3. These three were chosen from those available in SCIP based on preliminary experiments to determine which potential cuts would be added reasonably often and reasonably quickly. Each of these was also turned off in turn in order to assess whether it was positively contributing to the improved performance.

Six modified versions of the GOBNILP 1.3 resulted from this; three which each had one feature turned off and three which each had one type of cutting plane turned off. Each of the data sets was run on each of these systems and the time taken to find the optimal solution recorded. The results of these experiments are shown in Table 3.

The results show the biggest change in solution time occurs when the set packing constraints are removed. In nearly every case, this leads to an increase in solution time. In fact the situation can be even more extreme than the table suggests; for the `pigs 1000` data set, the system without the set packing constraints was allowed to continue running beyond the time out limit and had still not finished after more than 30 hours, when the full system finished in about 30 minutes.

Furthermore, in cases in which the two hour time limit was reached, the gap between the upper and lower bounds at that point was much larger in the version without set packing constraints than that for the full system. Closer examination revealed that this was because the score of the best BN found so far was the same as in the full system, but significantly less progress had been made in reducing the upper (dual) bound.

It is not immediately clear from this table if using the sinks heuristic aids the system or not. There are a number of problems for which it decreases solution

Network	m	p	N	Families	GOBNILP 1.3	GOBNILP 1.0	Cussens 2011	Without Solver Feature			No Cuts of Type			New VP
								SPC	SPH	VP	G	SCG	ZH	
			100	244	1	*3*	1	1	1	1	1	1	1	1
hailfinder	3	56	1000	761	5	*14*	5	*11*	5	**4**	**4**	**4**	**4**	5
			10000	3708	100	*361*	*169*	102	**56**	102	*558*	**75**	**83**	98
			100	4106	18	*270*		*34*	18	*34*	18	**13**	**13**	19
hailfinder	4	56	1000	767	4	*14*		*10*	4	4	*5*	*6*	4	4
			10000	4330	68	*934*		62	*128*	*587*	*216*	71	71	70
			100	907	2	*6*	*4*	*3*	2	**1**	2	2	2	2
alarm	3	37	1000	1928	5	*14*	*15*	*12*	**4**	5	*7*	**4**	5	**4**
			10000	6473	289	*792*	*12872*	*479*	*394*	*397*	*739*	*1049*	*710*	298
			100	1293	2	*7*		*9*	2	*7*	*3*	2	2	2
alarm	4	37	1000	2097	7	*15*		*21*	**6**	*8*	**6**	**3**	**6**	7
			10000	8445	398	*839*		*1253*	*806*	*1421*	*633*	*1567*	*1052*	398
			100	5068	756	*887*	*15176*	742	**628**	716	690	**642**	740	**651**
carpo	3	60	1000	3827	106	*171*	*593*	*143*	*134*	115	*117*	104	110	99
			10000	16391	1311	**566**	*42275*	*2158*	*1574*	**1071**	*4350*	**1032**	*2057*	1286
			100	13185	[0%]	[*1%*]		6649	[0%]	[0%]	[0%]	7014	[0%]	[0%]
carpo	4	60	1000	4722	151	*406*		*252*	*168*	*188*	*240*	*208*	140	149
			10000	34540	[0%]	4065		[0%]	[0%]	[0%]	[0%]	[0%]	[0%]	[0%]
			100	4441	2982	[*31%*]		[*39%*]	3082	3040	3036	**1506**	3212	2745
diabetes	2	413	1000	21493	[17%]	[*23%*]		[*168%*]	[*199%*]	[16%]	[17%]	[**15**%]	[17%]	[18%]
			10000	262129	[44%]	[**17**%]		[*378%*]	[*380%*]	[44%]	[44%]	[44%]	[44%]	[44%]
			100	2692	89	[*0%*]		*5103*	87	**32**	88	85	89	**32**
pigs	2	441	1000	15847	1818	[*7%*]		[*8%*]	1788	1715	1714	1802	1822	1657
			10000	304219	[3%]	[*9%*]		[*13%*]	[*42%*]	[3%]	[3%]	[3%]	[3%]	[3%]

Table 3: Comparison of GOBNILP 1.3 with older systems and impact of various features. p is the number of variables in the data set. m is the limit on the number of parents of each variable. N is the number of observations in the data set. Families is the number of family variables in the data set after pruning. All times are given in seconds (rounded). "[—]" indicates that the solution had not been found after 2 hours — the value given is the gap, rounded to the nearest percent, between the score of the best found BN and the upper bound on the score of the best potential BN, as a percentage of the score of the best found BN. Entries in italics are at least 10% worse than GOBNILP 1.3, while those in bold are at least 10% better. Key: SPC – Set Packing Constraints, SPH – Sink Primal Heuristic, VP – Value Propagator, G – Gomory cuts, SCG – Strong CG cuts, ZH – Zero-half cuts.

time and a number for which it increases it. For the most difficult problems, it appears to have little impact on solving time. As with the version without set packing constraints, the system without the primal heuristic also resulted in much larger gaps between the bounds in cases where it reached the time out limit. For the version without the primal heuristic, the difference is due to the best BN found so far being significantly worse after the time had elapsed, while the upper bound on the best possible BN that could be found was virtually identical to the full system. This latter point suggests that, while the sinks heuristic was of questionable value when solving problems to completion, on larger problems for which the algorithm may run out of time or resources before successfully finding the provably best BN, the heuristic plays an import role in ensuring that the best BN found so far is of high score.

The evidence for the effectiveness of the propagation is also mixed. In some cases, the system performs faster without the propagation, though study of the log files reveals this is almost exclusively down to the time spent directly carrying out propagation. Having noted this, a new faster propagator was created and used to replace the existing one. The result of running the full system with this faster propagator was to achieve a minor improvement over both the full system and the system without a propagator, as shown in the final column of Table 3.

The usefulness of each of the cutting plane algorithms is much clearer. Without Gomory cuts, the system is often slower and frequently by a large amount. On the other hand, removing Strong CG cuts usually improves the system's performance with one notable exception. Though checking log files, it can be confirmed that little time is spent generating Strong CG cuts and hence the improvement without them is due to a better search strategy rather than just a saving on the time spent adding these cuts. Zero-half cuts appear to have less effect than the other two studied but reduce solving time noticeably on two fairly large problems.

9 Conclusions and future work

Our principal conclusion is that IP is an attractive framework for exact BN learning from complete discrete data. However, as our comparative experiments demonstrate, some care (and empirical investigation) is required to properly exploit its potential. We have shown considerable advances over the results presented by Cussens [6] who himself presented faster solving times on four problems than [13]. However, it would clearly be desirable to compare GOBNILP 1.3 against further exact BN learning systems, not necessarily IP-based. We intend to compare against the URLearning system [17] in the immediate future.

In this paper we have focused on efficiently finding BNs with maximal score subject to constraints on parent set size. This raises the question of whether it is worth the effort to do this if one's ultimate goal is to return a DAG with high structural accuracy. In the context of 'pedigree reconstruction', work by Cussens *et al* [8] answers this question in the positive. In that paper an exact learning approach led to high structural accuracy. However, in a pedigree (a 'family tree') no node can have more than two parents. In other applications where the 'true' structure may have nodes with many parents our current restriction on parent set size may lead to poor structural accuracy. This is a clear limitation which we intend to address by the IP technique of *delayed column generation* where IP variables (i.e. parent sets) are created during solving [7].

In practical applications one often has prior knowledge concerning the (likely) structure of the 'true' BN. Because our GOBNILP 1.3 system is an example of *declarative machine learning* it is very easy to allow the user to declare constraints on BN structure when these can be encoded as linear constraints. Although we have not exploited it in the experiments reported here, GOBNILP 1.3 allows the user to declare the absence/presence of (i) particular directed edges, (ii) particular adjacencies and (iii) particular immoralities. In addition upper and lower bounds on the number of edges and founder nodes are possible. It is also possible to rule out specific BNs with a linear constraint. This allows GOBNILP 1.3 to not only find the optimal BN but also the top k BNs in decreasing order of score.

Acknowledgements

Thanks to Milan Studený, Raymond Hemmecke and David Haws for useful discussions concerning the imset representation. We would like to thank three anonymous referees for their valuable comments. This work has been supported by the UK Medical Research Council (Project Grant G1002312). This work benefited from a visit by the second author to the Helsinki Institute for Information Technology supported by the Finnish Centre of Excellence in Computational Inference Research (COIN).

References

[1] Tobias Achterberg. *Constraint Integer Programming.* PhD thesis, TU Berlin, July 2007.

[2] David Avis. A revised implementation of the reverse search vertex enumeration algorithm. In Gil

Kalai and Günter M. Ziegler, editors, *Polytopes Combinatorics and Computation*, volume 29 of *DMV Seminar*, pages 177–198. Birkhuser Basel, 2000.

[3] Egon Balas and Shu Ming Ng. On the set covering polytope: 1. All the facets with coefficients in {0,1,2}. *Mathematical Programming*, 43:57–69, 1989.

[4] Vacláv Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete Mathematics*, 4:305–337, 1973.

[5] James Cussens. Bayesian network learning by compiling to weighted MAX-SAT. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI 2008)*, pages 105–112, Helsinki, 2008. AUAI Press.

[6] James Cussens. Bayesian network learning with cutting planes. In Fabio G. Cozman and Avi Pfeffer, editors, *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI 2011)*, pages 153–160, Barcelona, 2011. AUAI Press.

[7] James Cussens. Column generation for exact BN learning: Work in progress. In *Proc. ECAI-2012 workshop on COmbining COnstraint solving with MIning and LEarning (CoCoMile 2012)*, 2012.

[8] James Cussens, Mark Bartlett, Elinor M. Jones, and Nuala A. Sheehan. Maximum likelihood pedigree reconstruction using integer linear programming. *Genetic Epidemiology*, 37(1):69–83, Janary 2013.

[9] Cassio de Campos and Qiang Ji. Properties of Bayesian Dirichlet scores to learn Bayesian network structures. In *AAAI-10*, pages 431–436, 2010.

[10] Michel X. Goemans and Leslie A. Hall. The strongest facets of the acyclic subgraph polytope are unknown. In *Integer Programming and Combinatorial Optimization*, volume 1084 of *Lectures Notes in Computer Science*, pages 415–429. Springer, 1996.

[11] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. On the acyclic subgraph polytope. *Mathematical Programming*, 33(1):28–42, 1985.

[12] Raymond Hemmecke, Silvia Lindner, and Milan Studený. Characteristic imsets for learning Bayesian network structure. *International Journal of Approximate Reasoning*, 53:1336–1349, 2012.

[13] Tommi Jaakkola, David Sontag, Amir Globerson, and Marina Meila. Learning Bayesian network structure using LP relaxations. In *Proceedings of 13th International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, volume 9, pages 358–365, 2010. Journal of Machine Learning Research Workshop and Conference Proceedings.

[14] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, 2009.

[15] Tomi Silander and Petri Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI-06)*, pages 445–45, 2006.

[16] Laurence A. Wolsey. *Integer Programming.* John Wiley, 1998.

[17] Changhe Yuan and Brandon Malone. An improved admissible heuristic for learning optimal Bayesian networks. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI-12)*, Catalina Island, CA, 2012.

Qualitative Possibilistic Mixed-Observable MDPs

Nicolas Drougard, Jean-Loup Farges,
Florent Teichteil-Königsbuch
Onera – The French Aerospace Lab
2 avenue Edouard Belin
31055 Toulouse Cedex 4, France

Didier Dubois
IRIT – Paul Sabatier University
118 route de Narbonne
31062 Toulouse Cedex 4, France

Abstract

Possibilistic and qualitative POMDPs (π-POMDPs) are counterparts of POMDPs used to model situations where the agent's initial belief or observation probabilities are imprecise due to lack of past experiences or insufficient data collection. However, like probabilistic POMDPs, optimally solving π-POMDPs is intractable: the finite belief state space exponentially grows with the number of system's states. In this paper, a possibilistic version of Mixed-Observable MDPs is presented to get around this issue: the complexity of solving π-POMDPs, some state variables of which are fully observable, can be then dramatically reduced. A value iteration algorithm for this new formulation under infinite horizon is next proposed and the optimality of the returned policy (for a specified criterion) is shown assuming the existence of a "stay" action in some goal states. Experimental work finally shows that this possibilistic model outperforms probabilistic POMDPs commonly used in robotics, for a target recognition problem where the agent's observations are imprecise.

1 INTRODUCTION

Markov Decision Processes (MDPs) define a useful formalism to express sequential decision problems under uncertainty [2]. Partially Observable MDPs (POMDPs) [15] are used to model situations in which an agent does not know directly the current state of the system: its decisions are based on a probability distribution over the state space. This distribution known as "belief" is updated at each stage $t \in \mathbb{N}$ of the process using the current observation. This update based on Bayes' rule needs perfect knowledge of the agent's initial belief and of the transition and observation probability distributions.

Consider situations where the agent totally ignores the system's initial state, for instance a robot that is for the first time in a room with an unknown exit location (initial belief) and has to find the exit and reach it. In practice, no experience can be repeated in order to extract a frequency of the exit's location. In this kind of situation, uncertainty is not due to a random fact, but to a lack of knowledge: no frequentist initial belief can be used to define the model. A uniform probability distribution is often chosen in order to assign the same mass to each state. This choice can be justified based on the subjective probability theory [5] (the probability distribution represents then an exchangeable bet) but subjective probabilities and observation frequencies are combined during the belief update.

In other cases, the agent may strongly believe that the exit is located in a wall as in the vast majority of rooms, but it still grants a very small probability p_ϵ to the fact that the exit may be a staircase in the middle of the room. Even if this is very unlikely to be the case, this second option must be taken into account in the belief, otherwise Bayes' rule cannot correctly update it if the exit is actually in the middle of the room. Eliciting p_ϵ without past experience is not obvious at all and does not rely on any rational reasons, yet it dramatically impacts the agent's policy. On the contrary, possibilistic uncertainty models allow the agent to elicit beliefs with imprecise unbiased knowledge.

The π-POMDP model is a possibilistic and qualitative counterpart of the probabilistic POMDP model [12]: it allows a formal modeling of total ignorance using a possibility distribution equal to 1 on all the states. This distribution means that all states are equally possible independently of how likely they are to happen (no necessary state). Moreover, consider robotic missions using visual perception: observations of the robot agent are outputs of image processing algorithms whose semantics (image correlation, object matching, class inference, etc.) is so complex that probabilities of occurrence are hard to rigorously extract. Finding qualitative estimates of their recognition performance is easier: the π-POMDP model only require qualitative data, thus it allows to construct the model without using more information than really available.

However, just like the probabilistic POMDP model, this possibilistic model faces the difficulty of computing an optimal policy. Indeed, the size of its belief space exponentially grows with the size of the state space, which prevents the use of π-POMDPs in practice. In situations where most state variables are fully observable, an alternative structuring of the model still allows to solve the problem: the possibilistic Mixed-Observable MDP model (π-MOMDP), which is the first contribution of this paper, indeed allows us to reason with beliefs over the partially observed states only. In this model borrowed from probabilistic POMDPs [9, 1], states are factorized into two sets of fully and partially observable state variables. Whereas, in probabilistic POMDPs, this factorized model permits to reason about smaller *continuous* belief subspaces and speed up α-vector operations, its impact is totally different in possibilistic POMDPs: it allows us to reduce the size of the *discrete* belief state space.

Our second contribution is a possibilistic value iteration algorithm for this extension, which exploits the hybrid structure of the belief space. This algorithm is derived from a π-MDP algorithm, Sabbadin's work update, whose optimality of the returned policy is proved for an infinite horizon criterion which is made explicit. It relies on intermediate "stay" actions that are needed to guarantee convergence of the algorithm but that vanish in the optimized policy for non goal states; they are the possibilistic counterparts of the discount factor in probabilistic POMDPs.

Finally, we experimentally demonstrate that in some situations π-MOMDPs outperform their probabilistic counterparts for instance on information-gathering robotic problems where the observation function resulting from complex image processing algorithms is not precisely known, as it is often the case in realistic applications. This result is significant for us, because roboticists commonly think that probabilistic POMDPs, and more generally Bayesian approaches, are first-choice reasoning models to solve sequential information-gathering missions. We prove in this paper that sometimes possibilistic uncertainty models perform better in practice.

2 BACKGROUND

The Markov Decision Process framework models situations in which the *system*, for instance the physical part of an agent and its environment, has a Markovian dynamic over time. The different possible states of the system are represented by the elements s of the finite state space $\mathcal{S}$. The initial system state is denoted by $s_0 \in \mathcal{S}$. At each stage of the process, modeled by non negative integers $t \in \mathbb{N}$, the decisional part of the agent can choose an *action* a in the finite set $\mathcal{A}$. The chosen action a_t determines the uncertainty over the future state s_{t+1} knowing the current state s_t.

2.1 Qualitative possibilistic MDPs

The work of Sabbadin [12] proposes a possibilistic counterpart of Markov Decision Processes. In this framework, the transition uncertainty is modeled as *qualitative possibility distributions* over $\mathcal{S}$. Let $\mathcal{L}$ be the *possibility scale i.e.* a finite and totally ordered set whose greatest element is denoted by $1_\mathcal{L}$ and the least element by $0_\mathcal{L}$ (classically $\mathcal{L}=\{0, \frac{1}{k}, \frac{2}{k}, \dots, \frac{k-1}{k}, 1\}$ with $k \in \mathbb{N}^*$). A qualitative possibility distribution over $\mathcal{S}$ is a function $\pi : \mathcal{S} \rightarrow \mathcal{L}$ such that $\max_s \pi(s) = 1_\mathcal{L}$ (possibilistic normalization), implying that at least one entirely possible state exists. Inequality $\pi(s) < \pi(s')$ means that state s' is more plausible than state s. This modeling needs less information than the probabilistic one since the plausibilities of events are "only" classified (in $\mathcal{L}$) but not quantified.

The transition function T^π is defined as follows: for each pair of states $(s, s') \in \mathcal{S}^2$ and action $a \in \mathcal{A}$, $T^\pi(s, a, s') = \pi\left(s' \mid s, a\right) \in \mathcal{L}$, the possibility of reaching the system state s' conditionned on the current state s and action a. Scale $\mathcal{L}$ serves as well to model the *preference* over states: function $\mu : \mathcal{S} \rightarrow \mathcal{L}$ models the agent's preferences. A π-MDP is then entirely defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{L}, T^\pi, \mu \rangle$.

A *policy* is a sequence $(\delta_t)_{t \geqslant 0}$ of *decision rules* $\delta : \mathcal{S} \rightarrow \mathcal{A}$ indexed by the stage of the process $t \in \mathbb{N}$: $\delta_t(s)$ is the action executed in state s at decision epoch t. We denote by Δ_p the set of all p-sized policies $(\delta_0, \dots, \delta_{p-1})$. Let $\tau = (s_1, \dots, s_p) \in \mathcal{S}^p$ be a p-sized trajectory and $(\delta) = (\delta_t)_{t=0}^{p-1}$ a p-sized policy. The set of all the p-sized trajectories is denoted by $\mathcal{T}_p$.

The sequence $(s_t)_{t \geqslant 0}$ is a Markov process: the possibility of the trajectory $\tau = (s_1, \dots, s_p)$ which starts from s_0 using $(\delta) \in \Delta_p$ is then

$$\Pi\left(\tau \mid s_0, (\delta)\right) = \min_{t=0}^{p-1} \pi\left(s_{t+1} \mid s_t, \delta_t(s_t)\right).$$

We define the preference of $\tau \in \mathcal{T}_p$ as the preference of the last state: $M(\tau) = \mu(s_p)$. As advised in [13] for problems in which there is no risk of being blocked in an unsatisfactory state, we use here the optimistic qualitative decision criterion [3] which is the Sugeno integral of the preference distribution over trajectories using possibility measure:

$$u_p\left(s_0, (\delta)\right) = \max_{\tau \in \mathcal{T}_p} \min\left\{\Pi\left(\tau \mid s_0, (\delta)\right), M(\tau)\right\}. \quad (1)$$

A policy which maximizes criterion 1 ensures that there exists a possible and satisfactory p-sized trajectory. The finite horizon π-MDP is solved when such a policy is found. The optimal p-sized horizon criterion $u_p^*(s) = \max_{(\delta)\in\Delta_p} u_p(s,(\delta))$ is the solution of the following dynamic programming equation, as proved in [6]: $\forall i \in \{1 \dots p\}$, $\forall s \in \mathcal{S}$,

$$u_i^*(s) = \max_{a\in\mathcal{A}} \max_{s'\in\mathcal{S}} \min \left\{ \pi\left(s' \mid s,a\right), u_{i-1}^*(s') \right\}, \quad (2)$$
$$\delta_{p-i}^*(s) \in \operatorname*{argmax}_{a\in\mathcal{A}} \max_{s'\in\mathcal{S}} \min \left\{ \pi\left(s' \mid s,a\right), u_{i-1}^*(s') \right\}$$

with the initialization $u_0^*(s) = \mu(s)$.

2.2 The partially observable case

A possibilistic counterpart of POMDPs is also given in [12]. As in the probabilistic framework, the agent does not directly observe the system's states. Here, uncertainty over observations is also modeled as possibility distributions. The observation function Ω^π is defined as follows: $\forall o' \in \mathcal{O}$, $s \in \mathcal{S}$ and $a \in \mathcal{A}$, $\Omega^\pi\left(s', a, o'\right) = \pi\left(o' \mid s, a\right)$ the possibility of the current observation o' conditionned on the current state s' and the previous action a. Then a π-POMDP is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{L}, T^\pi, \mathcal{O}, \Omega^\pi, \mu, \beta_0\rangle$, where β_0 is the initial possibilistic belief. The belief of the agent is a possibility distribution over $\mathcal{S}$; total ignorance is defined by a belief equal to $1_\mathcal{L}$ on all states, whereas a given state s is perfectly known if the belief is equal to $1_\mathcal{L}$ on this state and to $0_\mathcal{L}$ on all other states.

The translation into π-MDP can be done in a similar way as for POMDPs: we denote by $B^\pi \subset \mathcal{L}^\mathcal{S}$ the possibilistic belief state space which contains all the possibility distributions defined on $\mathcal{S}$. We can now compute the possibilistic belief update. If at time t the current belief is $\beta_t \in B^\pi$ and the agent executes action a_t, the belief over the future states is given by:

$$\beta_{t+1}^{a_t}(s') = \max_{s\in\mathcal{S}} \min \left\{ \pi\left(s' \mid s, a_t\right), \beta_t(s) \right\},$$

and the belief over the observations

$$\beta_{t+1}^{a_t}(o') = \max_{s'\in\mathcal{S}} \min \left\{ \pi\left(o' \mid s', a_t\right), \beta_{t+1}^{a_t}(s') \right\}.$$

Next, if the agent observes $o_{t+1} \in \mathcal{O}$, the possibilistic counterpart of Bayes' rule ensures that

$$\beta_{t+1}(s') = \begin{cases} 1_\mathcal{L} \text{ if } \beta_{t+1}^{a_t}(o_{t+1}) = \pi\left(s', o_{t+1} \mid \beta_t, a_t\right) > 0_\mathcal{L} \\ \pi\left(s', o_{t+1} \mid \beta_t, a_t\right) \text{ otherwise} \end{cases} \quad (3)$$

where $\forall (s',o') \in \mathcal{S} \times \mathcal{O}$, $\pi\left(s', o' \mid \beta, a\right) = \min\left\{\pi\left(o' \mid s', a\right), \beta^a(s')\right\}$ is the joint possibility of (s', o'). Such an update of a belief β is denoted by $\beta^{a,o'}$: $\beta_{t+1} = \beta_t^{a,o'}$. As the belief update is now made explicit, its dynamics can be computed: let $\Gamma^{\beta,a}\left(\beta'\right) = \left\{ o' \in \mathcal{O} \mid \beta^{a,o'} = \beta' \right\}$. Then $\pi\left(\beta' \mid \beta, a\right) = \max_{o' \in \Gamma^{\beta,a}(\beta')} \beta^a\left(o'\right)$ with the convention $\max_\emptyset = 0_\mathcal{L}$.

We now define the preference over belief states with a pessimistic form in order to prefer informative beliefs: a belief state has a good preference when it is unlikely that the system is in an unsatisfactory state: $\mu(\beta) = \min_{s\in\mathcal{S}} \max\left\{\mu(s), n(\beta(s))\right\}$, with $n : \mathcal{L} \to \mathcal{L}$ the order reversing map *i.e.* the only decreasing function from $\mathcal{L}$ to $\mathcal{L}$. A π-MDP is then defined, and the new dynamic programming equation is $\forall i \in \{1, \dots, p\}$, $\forall \beta \in B^\pi$,

$$\begin{aligned} u_p^*(\beta) &= \max_{a\in\mathcal{A}} \max_{\beta'\in B^\pi} \min\left\{\pi\left(\beta' \mid \beta, a\right), u_{p-1}^*(\beta')\right\} \\ &= \max_{a\in\mathcal{A}} \max_{o'\in\mathcal{O}} \min\left(\beta^a(o'), u_{p-1}^*(\beta^{a,o'})\right) \end{aligned}$$

with the initialization $u_0^*(\beta) = \mu(\beta)$. Note that B^π is a finite set of cardinal $\#B^\pi = \#\mathcal{L}^{\#\mathcal{S}} - (\#\mathcal{L} - 1)^{\#\mathcal{S}}$ (the total number of $\#\mathcal{S}$-size vectors valued in $\mathcal{L}$, minus $(\#\mathcal{L}-1)^{\#\mathcal{S}}$ non-normalized distributions). However, for concrete problems, the state space can be dramatically large: $\#B^\pi$ explodes and computations become intractable like in standard probabilistic POMDPs. The next section presents the first contribution of this paper, which exploits a specific structure of the problem that is very common in practice.

3 Possibilistic and qualitative mixed observable MDPs (π-MOMDPs)

The complexity issue of π-POMDP solving is due to the fact that the size of the belief state space B^π exponentially grows with the size of the state space $\mathcal{S}$. However, in practice, states are rarely totally hidden. Using mixed observability can be a solution: inspired by a similar recent work in probabilistic POMDPs [9, 1], we present in this section a structured modeling that takes into account situations where the agent directly observes some part of the state. This model generalizes both π-MDPs and π-POMDPs.

Like in [1], we assume that the state space $\mathcal{S}$ can be written as a Cartesian product of a visible state space $\mathcal{S}_v$ and a hidden one $\mathcal{S}_h$: $\mathcal{S} = \mathcal{S}_v \times \mathcal{S}_h$. Let $s = (s_v, s_h)$ be a state of the system. The component s_v is directly observed by the agent and s_h is only partially observed through the observations of the set $\mathcal{O}_h$: we

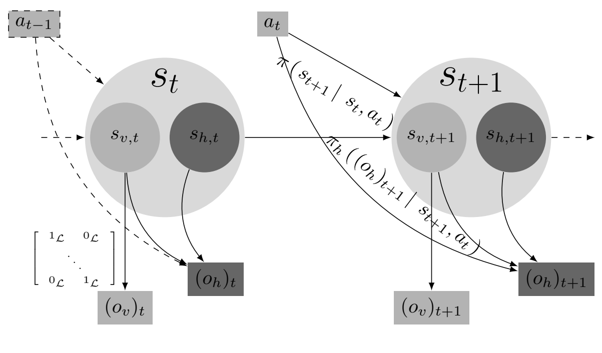

a_{t-1}
a_t
s_t
s_{t+1}
$s_{v,t}$
$s_{h,t}$
$s_{v,t+1}$
$s_{h,t+1}$
$\pi(s_{t+1} \mid s_t, a_t)$
$\pi_h((o_h)_{t+1} \mid s_{t+1}, a_t)$
$\begin{bmatrix} 1_\mathcal{L} & & 0_\mathcal{L} \\ & \ddots & \\ 0_\mathcal{L} & & 1_\mathcal{L} \end{bmatrix}$
$(o_v)_t$
$(o_h)_t$
$(o_v)_{t+1}$
$(o_h)_{t+1}$

Figure 1: Graphical representation of a π-MOMDP

denote by $\pi\left(o'_h \mid s', a\right)$ the possibility distribution over the future observation $o'_h \in \mathcal{O}_h$ knowing the future state $s' \in \mathcal{S}$ and the current action $a \in \mathcal{A}$. Figure 1 illustrates this structured model.

The visible state space is integrated to the observation space: $\mathcal{O}_v = \mathcal{S}_v$ and $\mathcal{O} = \mathcal{O}_v \times \mathcal{O}_h$. Then, knowing that the current visible component of the state is s_v, the agent *necessarily* observes $o_v = s_v$ (if $o'_v \neq s_v$, $\pi\left(o'_v \mid s_v\right) = 0_{\mathcal{L}}$). Formally, seen as a π-POMDP, its observation possibility distribution can be written as:

$$\begin{aligned}\pi\left(o' \mid s', a\right) &= \pi\left(o'_v, o'_h \mid s'_v, s'_h, a\right)\\ &= \min\left\{\pi\left(o'_h \mid s'_v, s'_h, a\right), \pi\left(o'_v \mid s'_v\right)\right\}\\ &= \left\{\begin{array}{cl}\pi\left(o'_h \mid s', a\right) & \text{if } o'_v = s'_v\\ 0_{\mathcal{L}} & \text{otherwise}\end{array}\right.\end{aligned} \quad (4)$$

since $\pi\left(o'_v \mid s'_v\right) = 1_{\mathcal{L}}$ if $s'_v = o'_v$ and $0_{\mathcal{L}}$ otherwise. The following theorem, based on this equality enables the belief over hidden states to be defined.

Theorem 1. *Each reachable belief state of a π-MOMDP can be written as an element of $\mathcal{S}_v \times B^\pi_h$ where B^π_h is the set of possibility distributions over $\mathcal{S}_h$: any $\beta \in B^\pi$ can be written as (s_v, β_h) with $\beta_h(s_h) = \max_{\overline{s}_v \in \mathcal{S}_v} \beta(\overline{s}_v, s_h)$ and $s_v = \operatorname{argmax}_{\overline{s}_v \in \mathcal{S}_v} \beta(\overline{s}_v, s_h)$.*

Proof. We proceed by induction on $t \in \mathbb{N}$: as the initial visible state $s_{v,0}$ is known by the agent, only states $s = (s_v, s_h)$ for which $s_v = s_{v,0}$ are such that $\beta_0(s) > 0_{\mathcal{L}}$. A belief over hidden states can be thus defined as $\beta_{h,0}(s_h) = \max_{s_v \in \mathcal{S}_v} \beta_0(s_v, s_h) = \beta_0(s_{v,0}, s_h)$.

At time t, if $\beta_t(s) = 0_{\mathcal{L}}$ for each $s = (s_v, s_h) \in \mathcal{S}$ such that $s_v \neq s_{v,t}$, the same notation can be adopted: $\beta_{h,t}(s_h) = \beta_t(s_{v,t}, s_h)$. Thus, if the agent reaches state $s_{t+1} = (s_{v,t+1}, s_{h,t+1})$ and if $s' = (s'_v, s'_h)$ with $s'_v \neq s_{v,t+1}$, then $s'_v \neq o_{v,t+1}$ and:

$$\begin{aligned}\pi\left(o_{t+1}, s' \mid \beta_t, a_t\right) &= \min\left\{\pi\left(o_{t+1} \mid s', a_t\right), \beta^{a_t}_{t+1}(s')\right\}\\ &= 0_{\mathcal{L}}.\end{aligned}$$

thanks to Equation (4). Finally, update Formula (3) ensures that $\beta_{t+1}(s') = 0_{\mathcal{L}}$. Then, β_{t+1} is entirely encoded by $(s_{v,t+1}, \beta_{h,t+1})$ with $s_{v,t+1} = o_{v,t+1}$ and $\beta_{h,t+1}(s_h) = \max_{s_v} \beta_{t+1}(s_v, s_h)\ \forall s_h \in \mathcal{S}_h$. □

As all needed belief states are in $\mathcal{S}_v \times B^\pi_h$, the next theorem redefines the dynamic programming equation restricted to this product space.

Theorem 2. *Over $\mathcal{S}_v \times B^\pi_h$, the dynamic programming equation becomes: $\forall i \in \{1, \ldots, p\}$, $\forall t \in \mathbb{N}$,*

$$u^*_i(s_v, \beta_h) = \max_{a \in \mathcal{A}} \max_{s'_v \in S_v} \max_{o'_h \in O_h} \min\left\{\beta^a(s'_v, o'_h), u^*_{i-1}(s'_v, \beta_h^{a,s'_v,o'_h})\right\}$$

with the initialization $\quad u^*_0(s_v, \beta_h) = \mu(s_v, \beta_h),$

where $\mu(s_v, \beta_h) = \min_{s_h \in \mathcal{S}_h} \max\left\{\mu(s_v, s_h), n(\beta_h(s_h))\right\}$ is the preference over $S_v \times B^\pi_h$,

$$\beta^a(s'_v, o'_h) = \max_{s'_h \in \mathcal{S}_h} \min\left\{\pi\left(o'_h \mid s'_v, s'_h, a\right), \beta^a(s'_v, s'_h)\right\},$$

and the belief update $\beta_h^{s'_v,o'_h,a}(s'_h)$

$$= \left\{\begin{array}{c}1_{\mathcal{L}} \text{ if } \min\left\{\pi\left(o'_h \mid s'_v, s'_h, a\right), \beta^a(s'_v, s'_h)\right\}\\ \qquad = \beta^a(s'_v, o'_h) > 0_{\mathcal{L}}\\ \min\{\pi\left(o'_h \mid s'_v, s'_h, a\right), \beta^a(s'_v, s'_h)\} \text{ otherwise}\end{array}\right.$$

Proof. Using the classical dynamic programming equation, Theorem 1, and the fact that $\mathcal{S}_v = \mathcal{O}_v$,

$$\begin{aligned}&u^*_i(s_v, \beta_h) = u^*_i(\beta)\\ &= \max_{a \in \mathcal{A}} \max_{(o'_v, o'_h) \in \mathcal{O}} \min\left\{\beta^a(o'_v, o'_h), u^*_{i-1}(\beta^{a,(o'_v,o'_h)})\right\}\\ &= \max_{a \in \mathcal{A}} \max_{s'_v \in \mathcal{S}_v} \max_{o'_h \in \mathcal{O}_h} \min\left\{\beta^a(s'_v, o'_h), u^*_{i-1}(\beta^{a,s'_v,o'_h})\right\}\\ &= \max_{a \in \mathcal{A}} \max_{s'_v \in \mathcal{S}_v} \max_{o'_h \in \mathcal{O}_h} \min\left\{\beta^a(s'_v, o'_h), u^*_{i-1}(s'_v, \beta_h^{a,s'_v,o'_h})\right\}\end{aligned}$$

where $\forall s_h \in \mathcal{S}_h$, $\beta_h^{a,s'_v,o'_h}(s_h) = \max_{\overline{s}_v} \beta^{a,s'_v,o'_h}(\overline{s}_v, s_h) = \beta^{a,s'_v,o'_h}(s'_v, s_h)$. For the initialization, we just note that $n(\beta(\overline{s}_v, s_h)) = 1_{\mathcal{L}}$ when $\overline{s}_v \neq s_v$, then

$$\begin{aligned}\mu(\beta) &= \min_{s} \max\left\{\mu(s), n(\beta(s))\right\}\\ &= \min_{s_h} \max\left\{\mu(s_v, s_h), n(\beta(s_h, s_v))\right\},\end{aligned}$$

which completes the procedure. The belief over observations defined in the last section can be written: $\forall o' = (o'_v, o'_h) \in \mathcal{O}$,

$$\begin{aligned}\beta^a(o') &= \max_{s' \in \mathcal{S}} \min\left\{\pi\left(o'_v, o'_h \mid s', a\right), \beta^{a_t}_t(s')\right\}\\ &= \max_{s'_h \in \mathcal{S}_h} \min\left\{\pi\left(o'_h \mid s'_v, s'_h, a_t\right), \beta^{a_t}_t(s'_v, s'_h)\right\}\end{aligned}$$

with $s'_v = o'_v$ since otherwise $\pi\left(o' \mid s'_v, s'_h, a\right) = 0_{\mathcal{L}}$ according to Equation (4). Then: $\beta^a(s'_v, o'_h) = \beta^a(o'_v, o'_h)$. Finally, using the standard update Equation (3) with $o'_v = s'_v$ and Equation (4), we get the new belief update. □

A standard algorithm would have computed $u^*_p(\beta)$ for each $\beta \in B^\pi$ while this new dynamic programming equation leads to an algorithm which computes it only for all $(s_v, \beta_h) \in \mathcal{S}_v \times B^\pi_h$. The size of the new belief space is $\#(\mathcal{S}_v \times B^\pi_h) = \#\mathcal{S}_v \times \left(\#\mathcal{L}^{\#\mathcal{S}_h} - (\#\mathcal{L} - 1)^{\#\mathcal{S}_h}\right)$, which is exponentially smaller than the size of standard π-POMDPs' belief space: $\#\mathcal{L}^{\#\mathcal{S}_v \times \#\mathcal{S}_h} - (\#\mathcal{L} - 1)^{\#\mathcal{S}_v \times \#\mathcal{S}_h}$.

4 Solving π-MOMDPs

A finite policy for possibilistic MOMDPs can now be computed for larger problems using the dynamic programming equation of Theorem 2 and selecting maximizing actions for each state $(s_v, \beta_h) \in \mathcal{S}_v \times B^\pi$, as done in Equation (2) for each $s \in \mathcal{S}$. However, for many problems in practice, it is difficult to determine a horizon size. The goal of this section is to present an algorithm to solve π-MOMDPs with infinite horizon, which is the first proved algorithm to solve π-(MO)MDPs.

4.1 The π-MDP case

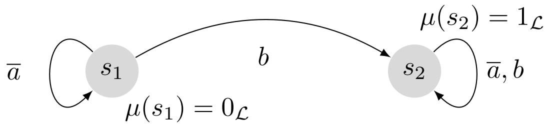
Previous work, [12, 14], on solving π-MDPs proposed a Value Iteration algorithm that was proved to compute optimal value functions, but not necessarily optimal policies for some problems with cycles. There is a similar issue in *undiscounted* probabilistic MDPs where the greedy policy at convergence of Value Iteration does not need to be optimal [11]. It is not surprising that we are facing the same issue in π-MDPs since the possibilistic dynamic programming operator does not rely on algebraic products so that it cannot be contracted by some *discount factor* $0 < \gamma < 1$.

Algorithm 1: π-MDP Value Iteration Algorithm

```
for $s \in \mathcal{S}$ do
    $u^*(s) \leftarrow 0_\mathcal{L}$ ;
    $u^c(s) \leftarrow \mu(s)$ ;
    $\delta(s) \leftarrow \overline{a}$ ;
while $u^* \neq u^c$ do
    $u^* = u^c$ ;
    for $s \in \mathcal{S}$ do
        $u^c(s) \leftarrow \max_{a \in \mathcal{A}} \max_{s' \in \mathcal{S}} \min\left\{ \pi\left(s' \mid s, a\right), u^*(s') \right\}$ ;
        if $u^c(s) > u^*(s)$ then
            $\delta(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \max_{s' \in \mathcal{S}} \min\left\{ \pi\left(s' \mid s, a\right), u^*(s') \right\}$ ;
return $u^*$, $\delta$ ;
```

To the best of our knowledge, we propose here the first Value Iteration algorithm for π-MDPs, that provably returns an optimal policy, and that is different from the one of [14]. Indeed, in the deterministic example of Figure 2, action $\overline{a}$, which is clearly suboptimal, was found to be optimal in state s_1 with this algorithm: however it is clear that since $\pi\left(s_2 \mid s_1, b\right) = 1_\mathcal{L}$ and $\mu(s_2) = 1_\mathcal{L}$, $u_1^*(s_1) = 1_\mathcal{L}$. Obviously, $u_1^*(s_2) = 1_\mathcal{L}$ and since $\pi\left(s_1 \mid s_1, \overline{a}\right) = 1_\mathcal{L}$, $\max_{s' \in \mathcal{S}} \min\left\{ \pi\left(s' \mid s_1, a\right), u_1^*(s') \right\} = 1_\mathcal{L}$ $\forall a \in \{\overline{a}, b\} = \mathcal{A}$, *i.e.* all actions are optimal in s_1. The "if" condition of Algorithm 1 permits to select the optimal action b during the first step. This condition and the initialization, which were not present in previous algorithms of the literature, are needed to prove the optimality of the policy. The proof, which is quite lengthy and intricate, is presented in Appendix A. This sound algorithm for π-MDPs will then be extended to π-MOMDPs in the next section.

Figure 2: Example



As mentioned in [12], we assume the existence of an action "stay", denoted by $\overline{a}$, which lets the system in the same state with necessity $1_\mathcal{L}$. This action is the possibilistic counterpart of the discount parameter γ in the probabilistic model, in order to guarantee convergence of the Value Iteration algorithm. However, we will see action $\overline{a}$ is finally used only on some particular satisfactory states. We denote by $\overline{\delta}$ is the decision rule such that $\forall s \in \mathcal{S}$, $\overline{\delta}(s) = \overline{a}$. The set of all the finite policies is $\Delta = \cup_{i \geqslant 1} \Delta_i$, and $\#\delta$ is the size of a policy (δ) in terms of decision epochs. We can now define the optimistic criterion for an infinite horizon: if $(\delta) \in \Delta$,

$$u(s, (\delta)) = \max_{\tau \in \mathcal{T}_{\#\delta}} \min\left\{ \Pi\left(\tau \mid s, (\delta)\right), M(\tau) \right\}. \quad (5)$$

Theorem 3. *If there exists an action $\overline{a}$ such that, for each $s \in \mathcal{S}$, $\pi\left(s' \mid s, \overline{a}\right) = 1_\mathcal{L}$ if $s' = s$ and $0_\mathcal{L}$ otherwise, then Algorithm 1 computes the maximum optimistic criterion and an optimal policy which is stationary i.e. which does not depend on the stage of the process t.*

Proof. See Appendix A. □

Let s be a state such that $\delta(s) = \overline{a}$, where δ is the returned policy. By looking at Algorithm 1, it can be noted that $u^*(s)$ always remains equal to $0_\mathcal{L}$ during the algorithm: $\forall s' \in \mathcal{S}$, either $\forall a \in \mathcal{A}$ $\mu(s) \geqslant \pi\left(s' \mid s, a\right)$, or $\mu(s) \geqslant u^*(s')$. If the problem is non trivial, it means that s is a goal ($\mu(s) > 0_\mathcal{L}$) and that degrees of possibility of transitions to better goals are less than the degree of preference for s.

4.2 Value Iteration for π-MOMDPs

We are now ready to propose the Value Iteration algorithm for π-MOMDPs. In order to clarify this algorithm, we set

$$U(a, s'_v, o'_h, \beta_h) = \min\left\{ \beta^a(s'_v, o'_h), u^*(s'_v, \beta_h^{a, s'_v, o'_h}) \right\}.$$

Note that Algorithm 2 has the same structure as Algorithm 1. Note that a π-MOMDP is a π-MDP over $\mathcal{S}_v \times B_h^\pi$. Let $s_v \in \mathcal{S}_v$, $\beta_h \in B_h^\pi$ and now $\Gamma_{\beta, \overline{a}, s'_v}(\beta'_h) = \left\{ o'_h \in \mathcal{O}_h \mid \beta_h^{\overline{a}, s'_v, o'_h} = \beta'_h \right\}$. To satisfy the assumption of Theorem 3, it suffices to ensure that $\max_{o'_h \in \Gamma_{\beta, \overline{a}, s'_v}(\beta'_h)} \beta^{\overline{a}}(s'_v, o'_h) = 1_\mathcal{L}$ if $s'_v = s_v$ and $\beta'_h = \beta_h$ and $0_\mathcal{L}$ otherwise. This property is verified when $\pi\left(s' \mid s, \overline{a}\right) = 1_\mathcal{L}$ if $s' = s$ (and $0_\mathcal{L}$ otherwise) and there exists an observation "nothing" $\overline{o}$ that is required for each state when $\overline{a}$ is chosen: $\pi\left(o' \mid s', \overline{a}\right) = 1_\mathcal{L}$ if $o' = \overline{o}$ and $0_\mathcal{L}$ otherwise.

Algorithm 2: π-MOMDP Value Iteration Algorithm

```
for $s_v \in \mathcal{S}_v$ and $\beta_h \in B_h^\pi$ do
    $u^*(s_v,\beta_h) \leftarrow 0_\mathcal{L}$ ;
    $u^c(s_v,\beta_h) \leftarrow \mu(s_v,\beta_h)$ ;
    $\delta(s_v,\beta_h) \leftarrow \overline{a}$ ;
while $u^* \neq u^c$ do
    $u^* = u^c$ ;
    for $s_v \in \mathcal{S}_v$ and $\beta_h \in B_h^\pi$ do
        $u^c(s) \leftarrow \max_{a \in \mathcal{A}} \max_{s'_v \in \mathcal{S}} \max_{o'_h \in \mathcal{O}_h} U(a, s'_v, o'_h, \beta_h)$ ;
        if $u^c(s_v,\beta_h) > u^*(s_v,\beta_h)$ then
            $\delta(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \max_{s'_v \in \mathcal{S}} \max_{o'_h \in \mathcal{O}_h} U(a, s'_v, o'_h, \beta_h)$ ;
return $u^*$, $\delta$ ;
```

5 Experimental results

Consider a robot over a grid of size $g \times g$, with $g > 1$. It always perfectly knows its location on the grid $(x, y) \in \{1, \ldots, g\}^2$, which forms the visible state space $\mathcal{S}_v$. It starts at location $s_{v,0} = (1, 1)$. Two targets are located at $(x_1, y_1) = (1, g)$ ("target 1") and $(x_2, y_2) = (g, 1)$ ("target 2") on the grid, and the robot perfectly knows their positions. One of the targets is A, the other B and the robot's mission is to identify and reach target A as soon as possible. The robot does not know which target is A: the two situations, "target 1 is A" ($A1$) and "target 2 is A" ($A2$), constitute the hidden state space $\mathcal{S}_h$. The moves of the robot are deterministic and its actions $\mathcal{A}$ consist in moving in the four directions plus the action "stay".
At each stage of the process, the robot analyzes pictures of each target and gets then an observation of the targets' natures: the two targets (oAA) can be observed as A, or target 1 (oAB), or target 2 (oBA) or no target (oBB).

In the probabilistic framework, the probability of having a good observation of target $i \in \{1, 2\}$, is not really known but approximated by $Pr(good_i \mid x, y) = \frac{1}{2}\left[1 + \exp\left(-\frac{\sqrt{(x-x_i)^2+(y-y_i)^2}}{D}\right)\right]$ where $(x, y) = s_v \in \{1, \ldots, g\}^2$ is the location of the robot, (x_i, y_i) the position of target i, and D a normalization constant. Then, for instance, $Pr\,(oAB \mid (x, y), A1)$ is equal to $Pr\,(good_1 \mid (x, y))\, Pr\,(good_2 \mid (x, y))$, $Pr\,(oAA \mid (x, y), A1)$ to $Pr\,(good_1 \mid (x, y)) \times [1 - Pr\,(good_2 \mid (x, y))]$, and so on. Each step of the process before reaching a target costs 1, reaching target A is rewarded by 100, and -100 for B. The probabilistic policy was computed in mixed-observability settings with APPL [9], using a precision of 0.046 (the memory limit is reached for higher precisions) and $\gamma = 0.99$. This problem can not be solved with the exact algorithm for MOMDPs [1] because it consumes the entire RAM after 15 iterations.

Using qualitative possibility theory, it is always possible to observe the good target: $\pi\,(good \mid x, y) = 1$. Here $\mathcal{L}$ will be a finite subset of $[0, 1]$, that is why $1_\mathcal{L}$ can be denoted by 1. Secondly, the more the robot is far away from target i, the more likely it can badly observe it (e.g. observe A instead of B), which is a reasonable assumption concerning the imprecisely known observation model: $\pi\,(bad_i \mid x, y) = \frac{\sqrt{(x-x_i)^2+(y-y_i)^2}}{\sqrt{2}(g-1)}$. Then for instance, $\pi\,(oAB \mid (x, y), A1) = 1$, $\pi\,(oAA \mid (x, y), A1) = \pi\,(bad_2 \mid x, y)$, $\pi(oBA \mid (x, y), A1) = \min\{\pi\,(bad_1 \mid x, y), \pi\,(bad_2 \mid x, y)\}$, etc. Note that the situation is fully known when the robot is at a target's location: thus there is no risk of being blocked in an unsatisfactory state, that is why using the *optimistic* π-MOMDP works. $\mathcal{L}$ thus consists in 0, 1, and all the other intermediate possible values of $\pi\,(bad \mid x, y)$. Note that the construction of this model with a probability-possibility transformation [4] would have been equivalent. The preference distribution μ is equal to 0 for all the system's states and to 1 for states $[(x_1, y_1), A1]$ and $[(x_2, y_2), A2]$ where (x_i, y_i) is the position of target i. As mentioned in [12], the computed policy guarantees a shortest path to a goal state. The policy then aims at reducing the mission's time.

Standard π-POMDPs, which do not exploit mixed observability contrary to our π-MOMDP model, could not solve even very small 3×3 grids. Indeed, for this problem, $\#\mathcal{L} = 5$, $\#\mathcal{S}_v = 9$, and $\#\mathcal{S}_h = 2$. Thus, $\#\mathcal{S} = \#\mathcal{S}_v \times \#\mathcal{S}_h = 18$ and the number of belief states is then $\#B^\pi = \mathcal{L}^{\#\mathcal{S}} - (\mathcal{L}^{\#\mathcal{S}} - 1)^{\#\mathcal{S}} = 5^{18} - 4^{18} \geqslant 3.7.10^{12}$ instead of 81 states with a π-MOMDP. Therefore, the following experimental results could **not** be conducted with standard π-POMDPs, which indeed justifies our present work on π-MOMDPs.

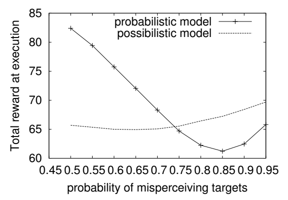
In order to compare performances of the probabilistic and possibilistic models, we compare *their total rewards at execution*: since the situation is fully known when the robot is at a target's location, it can not end up choosing target B. If k is the number of time steps to identify and reach the correct target, then the total reward is $100 - k$.

We consider now that, in reality (thus here for the simulations), used image processing algorithms badly perform when the robot is far away from targets, *i.e.*, if $\forall i \in \{1, 2\}$, $\sqrt{(x - x_i)^2 + (y - y_i)^2} > C$, with C a positive constant, $Pr\,(good_i \mid x, y) = 1 - P_{bad} < \frac{1}{2}$. In all other cases, we assume that the probabilistic model is the good one. We used 10^4 simulations to compute the statistical mean of the total reward at execution.

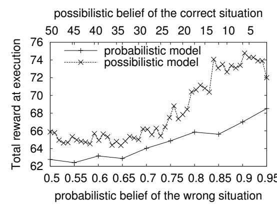
The grid was 10×10, $D = 10$ and $C = 4$.

Figure 3.a shows that the probabilistic is more affected by the introduced error than the possibilistic one: it shows the total reward at execution of each model as a function of P_{bad}, the probability of badly observing tagets when the robot's location is such that $\sqrt{(x - x_i)^2 + (y - x_i)^2} > C$. This is due to the fact that the possibilistic update of the belief does not take into account new observations when the robot has already obtained a more reliable one, whereas the probabilistic model modifies the current belief at each step. Indeed, as there are only two hidden states (that we now denote by s_h^1 and s_h^2), if $\beta_h(s_h^1) < 1$, then $\beta_h(s_h^2) = 1$ (possibilistic normalization). The definition of the joint possibility of a state and an observation (minimum of the belief in state and observation possibilities) imply that the joint possibility of s_h^1 and the obtained observation, is smaller than $\beta_h(s_h^1)$. The possibilistic counterpart of the belief update equation (3) then ensures that the next belief is either more skeptic about s_h^1 if the observation is more reliable and confirms the prior belief ($\pi\left(o_h \mid s_v, s_h^1, a\right)$ is smaller than $\beta_h(s_h^1)$); or changes to the opposite belief if the observation is more reliable and contradicts the prior belief ($\pi\left(o_h \mid s_v, s_h^2, a\right)$ is smaller than both $\beta_h(s_h^1)$ and $\pi\left(o_h \mid s_v, s_h^1, a\right)$); or yet simply remains unchanged if the observation is not more informative than the current belief. The probabilistic belief update does not have these capabilities to directly change to the opposite belief and to disregard less reliable observations: the robot then proceed towards the wrong target because it is initially far away and thus badly observes targets. When it is close to this target, it gets good observations and gradually modifies its belief which becomes true enough to convince it to go towards the right target. However it has to cross a remote area away from targets: this yet gradually modifies its belief, which becomes wrong, and the robot finds itself in the same initial situation: it loses thus a lot of time to get out of this loop. We can observe that the total reward increases for high probabilities of misperceiving P_{bad}: this is because this high error leads the robot to reach the wrong target faster, thus to entirely know that the true target is the other one.

Now if we set $P_{bad} = 0.8$ and evaluate the total reward at execution for different wrong initial beliefs, we get Figure 3.b with the same parameters: we compare here the possibilistic model and the probabilistic one when the initial belief is strongly oriented towards the wrong hidden states (i.e. the agent strongly believes that target 1 is B whereas it is A in reality). Note that the possibilistic belief of the good target decreases when the necessity of the bad one increases. This figure shows that the possibilistic model yields



(a) Varying P_{bad}.



(b) Varying β_0.

Figure 3: Comparison of the total reward gathered at execution for possibilistic and probabilistic models.

higher rewards at execution if the initial belief is wrong and the observation function is imprecise.

6 Conclusion and perspectives

We have proposed a Value Iteration algorithm for possibilistic MDPs, which can produce optimal stationary policies in infinite horizon contrary to previous methods. We have provided a complete proof of convergence that relies on the existence of intermediate "stay" actions that vanish for non goal states in the final optimal policy. Finally, we have extended this algorithm to a new Mixed-Observable possibilistic MDP model, whose complexity is exponentially smaller than possibilistic POMDPs, so that we could compare π-MOMDPs with their probabilistic counterparts on realistic robotic problems. Our experimental results show that possibilistic policies can outperform probabilistic ones when the observation function yields imprecise results.

Qualitative possibilistic frameworks can however be inappropriate when some probabilistic information is actually available: POMDPs with Imprecise Parameters (POMDPIP) [7] and Bounded-parameter POMDPs (BPOMDP) [8] integrate the lack of knowledge by considering spaces of possible probability distributions. When such spaces can not be extracted or when a qualitative modeling suffices, π-POMDPs can be a good alternative, especially as POMDPIPs and BPOMDPs are extremely difficult to solve in practice. Yet, we plan to compare our π-MOMDP model with these imprecise probabilistic POMDPs in a near future.

The pessimistic version of π-MDPs can be easily constructed, but the optimality of the policy returned by the associated value iteration algorithm seems hard to prove, essentially because it is not enough to construct a maximizing trajectory, as the proof of section A does. The works [16, 10] could help us to get results about pessimistic π-MDP in order to solve unsafe problems.

A Proof of Theorem 3

This appendix demonstrates that Algorithm 1 returns the maximum value of Equation (5) and an optimal policy. Note that the policy is optimal regardless of the initial state. We recall that $\exists \overline{a} \in \mathcal{A}$ such that $\forall s \in \mathcal{S}$, $\pi\left(s' \mid s, \overline{a}\right) = 1_{\mathcal{L}}$ if $s' = s$, and $0_{\mathcal{L}}$ otherwise. The existence of this action $\overline{a}$ makes the maximum value of the criterion non-decreasing with respect to the horizon size:

Lemma 1. $\forall s \in \mathcal{S}$, $\forall p \geqslant 0$, $u^*_p(s) \leqslant u^*_{p+1}(s)$.

Proof. Let $s_0 \in \mathcal{S}$. $u^*_{p+1}(s_0)$
$= \max\limits_{\Delta_{p+1}} \max\limits_{\tau \in \mathcal{T}_{p+1}} \min\left\{ \min\limits_{i=0}^{p} \pi\left(s_{i+1} \mid s_i, \delta_i(s_i)\right), \mu(s_{p+1}) \right\}$.
Consider the particular trajectories $\tau' \in \mathcal{T}'_{p+1} \subset \mathcal{T}_{p+1}$ such that $\tau' = (s_1, \ldots, s_p, s_p)$, and particular policies $(\delta') \in \Delta'_{p+1} \subset \Delta_{p+1}$ such that $(\delta') = (\delta_0, \ldots, \delta_{p-1}, \overline{\delta})$. It is obvious that $u^*_{p+1}(s_0) \geqslant$
$\max\limits_{(\delta') \in \Delta'_{p+1}} \max\limits_{\tau' \in \mathcal{T}'_{p+1}} \min\left\{ \min\limits_{i=0}^{p} \pi\left(s_{i+1} \mid s_i, \delta_i(s_i)\right), \mu(s_{p+1}) \right\}$.
But note that the right part of this inequality can be rewritten as $\max\limits_{(\delta) \in \Delta_p} \max\limits_{\tau \in \mathcal{T}_p}$ $\min\left\{ \min_{i=0}^{p-1} \pi\left(s_{i+1} \mid s_i, \delta_i(s_i)\right), \pi\left(s_p \mid s_p, \overline{a}\right), \mu(s_p) \right\}$
$= u^*_p(s_0)$ since $\pi\left(s_p \mid s_p, \overline{a}\right) = 1_{\mathcal{L}}$. □

The meaning of this lemma is: it is always more possible to reach a state s from s_0 in at most $p+1$ steps than in at most p steps. As for each $s \in \mathcal{S}$, $(u^*_p(s))_{p \in \mathbb{N}} \leqslant 1_{\mathcal{L}}$, Lemma 1 insures that the sequence $(u^*_p(s))_{p \in \mathbb{N}}$ converges. The next lemma shows that the convergence of this sequence occurs in finite time.

Lemma 2. *For all $\forall s \in \mathcal{S}$, the number of iterations of the sequence $(u^*_p(s))_{p \in \mathbb{N}}$ up to convergence is bounded by $\#\mathcal{S} \times \#\mathcal{L}$.*

Proof. Recall first that values of the possibility and preference distributions are in $\mathcal{L}$ which is finite and totally ordered: we can write $\mathcal{L}=\{0_{\mathcal{L}}, l_1, l_2, \ldots, 1_{\mathcal{L}}\}$ with $0_{\mathcal{L}} < l_1 < l_2 < \ldots < 1_{\mathcal{L}}$. If two successive functions u^*_k and u^*_{k+1} are equal, then $\forall s \in \mathcal{S}$ sequences $(u^*(s))_{p \geqslant k}$ are constant. Indeed this sequence can be defined by the recursive formula

$$u^*_p(s) = \max_{a \in \mathcal{A}} \max_{s' \in \mathcal{S}} \min\left\{ \pi\left(s' \mid s, a\right), u^*_{p-1}(s') \right\}.$$

Thus if $\forall s \in \mathcal{S}$, $u^*_p(s) = u^*_{p-1}(s)$ then the next iteration $(p+1)$ faces the same situation ($u^*_{p+1}(s) = u^*_p(s)$ $\forall s \in \mathcal{S}$). The slowest convergence can then be described as follows: for each $p \in \mathbb{N}$ only one $s \in \mathcal{S}$ is such that $u^*_{p+1}(s) > u^*_p(s)$. Moreover, for this s, if $u^*_p(s) = l_i$, then $u^*_{p+1}(s) = l_{i+1}$. We can conclude that for $p > \#\mathcal{L} \times \#\mathcal{S}$, the sequence is constant. □

First note that the variable $u^*(s)$ of Algorithm 1 is equal to $u^*_p(s)$ after the p^{th} iteration. We conclude that u^* converges to the maximal value of the criterion for an $(\#\mathcal{L} \times \#\mathcal{S})$-size horizon and can not be greater: the function u^* returned is thus optimal with respect to Equation (5) and is computed in a finite number of steps.

In the following, we prove the optimality of the policy (δ^*) returned by Algorithm 1. For this purpose, we will construct a trajectory of size smaller than $\#\mathcal{S}$ which maximizes $\min\left\{ \Pi\left(\tau \mid s_0, (\delta)\right), M(\tau) \right\}$ with policy (δ^*). The next two lemmas are needed for this construction and require some notations.

Let $s_0 \in \mathcal{S}$ and p be the smallest integer such that $\forall p' \geqslant p$, $u^*_{p'}(s_0) = u^*(s_0)$, where u^* is here the optimal value of the infinite horizon criterion of Equation (5) (variable $u^*(s)$ of Algorithm 1 does not increase after p iterations). Equation (2) can be used to return an optimal policy (not stationary) denoted by $(\delta^{(s_0)}) \in \Delta_p$. With this notation: $\forall s \in \mathcal{S}$, $\delta^*(s) = \delta^{(s)}_0(s)$. Consider now a trajectory $\tau = (s_1, s_2, \ldots, s_p)$ which maximizes $\min\left\{ \min_{i=0}^{p-1} \pi\left(s_{i+1} \mid s_i, \delta^{(s_0)}_i(s_i)\right), \mu(s_p) \right\}$. This trajectory is called *optimal trajectory of minimal size from* s_0.

Lemma 3. *Let $\tau = (s_1, \ldots, s_p)$ be an optimal trajectory of minimal size from s_0.*
Then, $\forall k \in \{1, \ldots, p-1\}$, $u^*(s_0) \leqslant u^*(s_k)$.

Proof. Let $k \in \{1, \ldots, p-1\}$.

$$\begin{aligned} u^*(s_0) &= \min\left\{ \min_{i=0}^{p-1} \pi\left(s_{i+1} \mid s_i, \delta^{(s_0)}_i(s_i)\right), \mu(s_p) \right\} \\ &\leqslant \min\left\{ \min_{i=k}^{p-1} \pi\left(s_{i+1} \mid s_i, \delta^{(s_0)}_i(s_i)\right), \mu(s_p) \right\} \end{aligned}$$

$\leqslant u^*_{p-k}(s_k) \leqslant u^*(s_k)$ since $(u^*_p)_{p \in \mathbb{N}}$ is non-decreasing (Lemma 1). □

Lemma 4. *Let $\tau = (s_1, \ldots, s_p)$ be an optimal trajectory of minimal size from s_0 and $k \in \{1, \ldots, p-1\}$. If $u^*(s_0) = u^*(s_k)$, then $\delta^*(s_k) = \delta^{(s_0)}_k(s_k)$.*

Proof. Suppose that $u^*(s_0) = u^*(s_k)$. Since $u^*(s_0) \leqslant u^*_{p-k}(s_k) \leqslant u^*(s_k)$ (Lemma 3), we obtain that $u^*_{p-k}(s_k) = u^*(s_k)$. The criterion in s_k is thus optimized within a $(p-k)$-horizon. Moreover a shorter horizon is not optimal: $\forall m \in \{1, \ldots, p-k\}$, $u^*_{p-k-m}(s_k) < u^*(s_k)$ *i.e.* with a $(p-k-m)$-size horizon the criterion in s_k is not maximized. Indeed if the contrary was true, the criterion in s_0 would be maximized within a $(p-m)$-size horizon: the policy

$$\delta' = (\delta^{(s_0)}_0, \delta^{(s_0)}_1, \ldots, \delta^{(s_0)}_{k-1}, \delta^{(s_k)}_0, \ldots, \delta^{(s_k)}_{p-k-m-1}) \in \Delta_{p-m}$$

would be optimal. Indeed, $u^*(s_0)$

$$\begin{aligned} &= \min\left\{ \min_{i=0}^{k-1} \pi\left(s_{i+1} \mid s_i, \delta^{(s_0)}_i(s_i)\right), u^*_{p-k}(s_k) \right\} \\ &= \min\left\{ \min_{i=0}^{k-1} \pi\left(s_{i+1} \mid s_i, \delta^{(s_0)}_i(s_i)\right), u^*(s_k) \right\} \end{aligned}$$

Then let $\overline{\tau} = (\overline{s}_1, \dots, \overline{s}_{p-k-m}) \in \mathcal{T}_{p-k-m}$ be an optimal trajectory of minimal size from s_k. Setting $\overline{s_0} = s_k$, $\overline{\tau}$ thus maximizes $u^*(s_k) = \min\left\{ \min_{i=0}^{p-k-m-1} \pi\left(\overline{s}_{i+1} \mid \overline{s}_i, \delta_i^{(s_k)}(\overline{s}_i) \right), \mu(\overline{s}_{p-k-m}) \right\}$. If $(s'_1, \dots, s'_{p-m}) = (s_1, \dots, s_{k-1}, \overline{s}_0, \dots, \overline{s}_{p-k-m})$, $u^*(s_0) = \min\left\{ \min_{i=0}^{p-m-1} \pi\left(s'_{i+1} \mid s'_i, \delta'_i(s_i) \right), \mu(s'_{p-m}) \right\}$ *i.e.* $\exists p' < p$ such that $u^*(s_0) = u^*_{p'}(s_0)$: it contradicts the assumption that $(s_1, \dots, s_p)$ is an optimal trajectory of minimum size. Thus $p-k$ is the smallest integer such that $u^*_{p-k}(s_k) = u^*(s_k)$: we finally conclude that $\delta^*(s_k)$ $(:= \delta_0^{(s_k)}(s_k))$ $= \delta_k^{(s_0)}(s_k)$. $\square$

Theorem 4. *Let (δ^*) be the policy returned by Algorithm 1; $\forall s_0 \in \mathcal{S}$, there exists $p^* \leqslant \#\mathcal{S}$ and a trajectory $(s_1, \dots, s_{p^*})$ such that*

$$u^*(s_0) = \min\left\{ \min_{i=0}^{p^*-1} \pi\left(s_{i+1} \mid s_i, \delta^*(s_i) \right), \mu(s_{p^*}) \right\}:$$

i.e. δ^* is an optimal policy.

Proof. Let s_0 be in $\mathcal{S}$ and τ be an optimal trajectory of minimal size p from s_0. If $\forall k \in \{1, \dots, p-1\}$, $\delta^*(s_k) := \delta_0^{(s_k)}(s_k) = \delta_k^{(s_0)}(s_k)$ then the criterion in s_0 is maximized with (δ^*) since it is maximized with $(\delta^{(s_0)})$ and the optimality is shown. If not, let k be the smallest integer $\in \{1, \dots, p-1\}$ such that $\delta_0^{(s_k)}(s_k) \neq \delta_k^{(s_0)}(s_k)$. Lemmas 3 and 4 ensure that $u^*(s_k) > u^*(s_0)$. Definition of k ensures that $u^*(s_k) > u^*(s_i)$ $\forall i \in \{0, \dots, k-1\}$.

Reiterate beginning with $s_0^{(1)} = s_k$: let $p^{(1)}$ be the number of iterations until variable $u^*(s^{(1)})$ of the algorithm converges (the smallest integer such that $u^*(s_0^{(1)}) = u^*_{p^{(1)}}(s_0^{(1)})$). Let $\tau^{(1)} \in T_{p^{(1)}}$ which maximizes $\min\{\min_{i=0}^{p^{(1)}-1} \pi(s_{i+1}|s_i, \delta_i^{(s_0^{(1)})}(s_i)), \mu(s_p^{(1)})\}$ ($\tau^{(1)}$ is an optimal trajectory of minimal size from $s_k = s_0^{(1)}$). We select $k^{(1)}$ in the same way as previously and reiterate beginning with $s_0^{(2)} = s_{k^{(1)}}^{(1)}$ which is such that $u^*(s_{k^{(1)}}^{(1)}) > u^*(s_0^{(1)})$, and $u^*(s_{k^{(1)}}^{(1)}) > u^*(s_i^{(1)})$ $\forall i \in \{0, \dots, k^{(1)}-1\}$ etc... Lemma 5 below shows that all selected states $(s_0, \dots, s_{k-1}, s_0^{(1)}, \dots, s_{k^{(1)}-1}^{(1)}, s_0^{(2)} \dots, s_{k^{(2)}-1}^{(2)}, s_0^{(3)}, \dots)$, are different. Thus this selection process ends since $\#\mathcal{S}$ is a finite set. The total number of selected states is denoted by $p^* = k + \sum_{i=1}^{q-1} k^{(i)} + p^{(q)}$ with $q \geqslant 0$ the number of new selected trajectories. Then the policy $(\delta') = (\delta_0, \dots, \delta_{k-1}, \delta_0^{(s_0^{(1)})}, \dots, \delta_{k^{(1)}-1}^{(s_0^{(1)})}, \dots, \delta_{p^{(q)}}^{(s_0^{(q)})})$ corresponds to (δ^*) over $\tau' = (s'_1, \dots, s'_{p^*})$ $= (s_0, s_1, \dots, s_{k-1}, s_0^{(1)}, \dots, s_{k^{(1)}-1}^{(1)}, \dots, s_{p^{(q)}-1}^{(m)})$ and this policy is optimal because $u^*(s_0) = u(s_0, (\delta^*))$:

$$u^*(s_0) = \min\left\{ \min_{i=0}^{k-1} \pi\left(s'_{i+1} \mid s'_i, \delta'(s'_i) \right), u^*_{p-k}(s_k) \right\}$$
$$\leqslant \min\left\{ \min_{i=0}^{k-1} \pi\left(s'_{i+1} \mid s'_i, \delta'(s'_i) \right), u^*(s_k) \right\}$$
$$= \min\left\{ \min_{i=0}^{k^{(1)}-1} \pi\left(s'_{i+1} \mid s'_i, \delta'(s'_i) \right), u^*_{p^{(1)}-k^{(1)}}(s_{k^{(1)}}) \right\}$$
$$\dots \leqslant \min\left\{ \min_{i=0,\dots,p^*-1} \pi\left(s'_{i+1} \mid s'_i, \delta'(s'_i) \right), \mu(s'_{p^*}) \right\}$$

The "$\leqslant$" signs are in fact "$=$" since otherwise we would find a policy such that $u(s_0, (\delta')) > u^*(s_0)$. Thus (δ^*) is optimal: $u^*(s_0) = \min\left\{ \min_{i=0}^{p^*-1} \pi\left(s'_{i+1} \mid s'_i, \delta^*(s'_i) \right), \mu(s'_{p^*}) \right\}$ $\square$

Lemma 5. *The process described in the previous proof in order to construct a trajectory maximizing the criterion with (δ^*) always selects different system states.*

Proof. First, two equal states in the same selected trajectory $\tau^{(m)}$ would contradict the hypothesis that $p^{(m)}$ is the smallest integer such that $u^*_{p^{(m)}}(s_0^{(m)}) = u^*(s_0^{(m)})$. Indeed let k and l be such that $0 \leqslant k < l \leqslant p^{(m)}$ and suppose that $s_k^{(m)} = s_l^{(m)}$. For clarity in the next calculations, we omit "(m)": $p = p^{(m)}$ and $\forall i \in \{0, \dots, l\}$, $s_i = s_i^{(m)}$. $u^*_{p-k}(s_k) = \min\left\{ \min_{i=k}^{l-1} \pi\left(s_{i+1} \mid s_i, \delta_i^{(s_0)}(s_i) \right), u^*_{p-l}(s_l) \right\}$ $\leqslant u^*_{p-l}(s_l) = u^*_{p-l}(s_k)$. However $u^*_{p-k}(s_k) \geqslant u^*_{p-l}(s_k)$ (non-decreasing sequence).
We finally get $u^*_{p-k}(s_k) = u^*_{p-l}(s_k)$, thus

$$u^*(s_0) = \min\left\{ \min_{i=0}^{k-1} \pi\left(s_{i+1} \mid s_i, \delta_i^{(s_0)}(s_i) \right), u^*_{p-k}(s_k) \right\}$$
$$= \min\left\{ \min_{i=0}^{k-1} \pi\left(s_{i+1} \mid s_i, \delta_i^{(s_0)}(s_i) \right), u^*_{p-l}(s_l) \right\}$$
$$= \min\left\{ \min_{i=0,\dots,k-1,l,\dots,p-1} \pi\left(s_{i+1} \mid s_i, \delta_i^{(s_0)}(s_i) \right), \mu(s_p) \right\}$$

Consequently, a $(p^{(m)} - l + k)$-sized horizon is good enough to reach the optimal value: it is a contradiction. Finally, if we suppose that a state $\overline{s}$ appears two times in the sequence of selected states, then this state belongs to two different selected trajectories $\tau^{(m)}$ and $\tau^{(m')}$ (with $m' < m$). Lemma 3 and the definition of $k^{(m')}$ which implies that $u^*(s_0^{(m'+1)})$ is strictly greater than the criterion's optimal values in each of the states $s_0^{(m')}, \dots, s_{k^{(m')}-1}^{(m')}$ requires that $u^*(s_0^{(m)}) \leqslant u^*(\overline{s}) < u^*(s_0^{(m'+1)})$. It is a contradiction because $u^*(s_0^{(m'+1)}) \leqslant u^*(s_0^{(m)})$ since $m' < m$. $\square$

References

[1] M. Araya-Lòpez, V. Thomas, O. Buffet, and F. Charpillet. A closer look at MOMDPs. In *Proceedings of the Twenty-Second IEEE International Conference on Tools with Artificial Intelligence (ICTAI-10)*, 2010.

[2] Richard Bellman. A Markovian Decision Process. *Indiana Univ. Math. J.*, 6:679–684, 1957.

[3] Didier Dubois and Henri Prade. Possibility theory as a basis for qualitative decision theory. pages 1924–1930. Morgan Kaufmann, 1995.

[4] Didier Dubois, Henri Prade, and Sandra Sandri. On possibility/probability transformations. In *Proceedings of Fourth IFSA Conference*, pages 103–112. Kluwer Academic Publ, 1993.

[5] Didier Dubois, Henri Prade, and Philippe Smets. Representing partial ignorance. *IEEE Trans. on Systems, Man and Cybernetics*, 26:361–377, 1996.

[6] Hélène Fargier, Jérôme Lang, and Régis Sabbadin. Towards qualitative approaches to multistage decision making . 19:441–471, 1998. FargLSab001.

[7] Hideaki Itoh and Kiyohiko Nakamura. Partially observable markov decision processes with imprecise parameters. *Artificial Intelligence*, 171(89):453 – 490, 2007.

[8] Yaodong Ni and Zhi-Qiang Liu. Policy iteration for bounded-parameter pomdps. *Soft Computing*, pages 1–12, 2012.

[9] Sylvie C. W. Ong, Shao Wei Png, David Hsu, and Wee Sun Lee. Planning under uncertainty for robotic tasks with mixed observability. *Int. J. Rob. Res.*, 29(8):1053–1068, July 2010.

[10] Cédric Pralet, Thomas Schiex, and Gérard Verfaillie. *Sequential Decision-Making Problems - Representation and Solution.* Wiley, 2009.

[11] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

[12] Régis Sabbadin. A possibilistic model for qualitative sequential decision problems under uncertainty in partially observable environments. In *15th Conference on Uncertainty in Artificial Intelligence (UAI99) , Stockholm, 30/07/99-01/08/99*, pages 567–564, San Francisco, juillet 1999. Morgan Kaufmann.

[13] Régis Sabbadin. Empirical comparison of probabilistic and possibilistic markov decision processes algorithms. In Werner Horn, editor, *ECAI*, pages 586–590. IOS Press, 2000.

[14] Régis Sabbadin. Possibilistic markov decision processes. *Engineering Applications of Artificial Intelligence*, 14(3):287 – 300, 2001. Soft Computing for Planning and Scheduling.

[15] Richard D. Smallwood and Edward J. Sondik. *The Optimal Control of Partially Observable Markov Processes Over a Finite Horizon*, volume 21. INFORMS, 1973.

[16] Paul Weng. Conditions générales pour l'admissibilité de la programmation dynamique dans la décision séquentielle possibiliste. *Revue d'Intelligence Artificielle*, 21(1):129–143, 2007. NAT LIP6 DECISION.

Optimization With Parity Constraints: From Binary Codes to Discrete Integration[*]

Stefano Ermon, Carla P. Gomes
Department of Computer Science
Cornell University, Ithaca, NY, USA
{ermonste,gomes}@cs.cornell.edu

Ashish Sabharwal
IBM Watson Research Center
Yorktown Heights, NY, USA
ashish.sabharwal@us.ibm.com

Bart Selman
Dept. of Computer Science
Cornell University, Ithaca, USA
selman@cs.cornell.edu

Abstract

Many probabilistic inference tasks involve summations over exponentially large sets. Recently, it has been shown that these problems can be reduced to solving a polynomial number of MAP inference queries for a model augmented with randomly generated parity constraints. By exploiting a connection with max-likelihood decoding of binary codes, we show that these optimizations are computationally hard. Inspired by iterative message passing decoding algorithms, we propose an Integer Linear Programming (ILP) formulation for the problem, enhanced with new sparsification techniques to improve decoding performance. By solving the ILP through a sequence of LP relaxations, we get both lower and upper bounds on the partition function, which hold with high probability and are much tighter than those obtained with variational methods.

1 INTRODUCTION

Discrete probabilistic graphical models [18, 31] are often defined up to a normalization factor involving a summation over an exponentially large combinatorial space. Computing these factors is an important problem, as they are needed, for instance, to evaluate the probability of evidence, rank two alternative models, and learn parameters from data. Unfortunately, computing these discrete integrals exactly in very high dimensional spaces quickly becomes intractable, and approximation techniques are often needed. Among them, sampling and variational methods are the most popular approaches. Variational inference problems are typically solved using message passing techniques, which are often guaranteed to converge to some local minimum [30, 31], but without guarantees on the quality of the solution found. Markov Chain Monte Carlo [17, 21, 32] and Importance Sampling techniques [10, 11, 13] are asymptotically correct, but the number of samples required to obtain a statistically reliable estimate can grow exponentially in the worst case.

Recently, Ermon et al. [6] introduced a new technique called WISH which comes with provable (probabilistic) guarantees on the approximation error. Their method combines combinatorial optimization techniques with the use of universal hash functions to uniformly partition a large combinatorial space, originally introduced by Valiant and Vazirani to study the Unique Satisfiability problem and later exploited by Gomes et al. [13, 14] for solution counting. Specifically, they show that one can obtain the intractable normalization constant (partition function) of a graphical model within any desired degree of accuracy, by solving a polynomial number of MAP queries for the original graphical model augmented with randomly generated parity constraints as evidence. Although MAP inference is NP-hard and thus also intractable, this is a significant step forward as counting problems such as estimating the partition function are #-P hard, a complexity class believed to be significantly harder than NP.

In this work, we investigate the class of MAP inference queries with random parity constraints arising from the WISH scheme. These optimization problems turn out to be intimately connected with the fundamental problem of maximum likelihood decoding of a binary code [3, 29]. We leverage this connection to show that the inference queries generated by WISH are NP-hard to solve and to approximate, even for very simple graphical models. Although generally hard in the worst case, message passing and related linear programming techniques [7] are known to be very successful in practice in decoding certain types of codes such as low density parity check (LDPC) codes [8].

[*] This work was supported by NSF Grant 0832782.

Inspired by the success of these methods, we formulate the MAP inference queries generated by WISH as Integer Linear Programs (ILP). Unfortunately, such queries are typically harder than traditional decoding problems because they involve more complex probabilistic models, and because universal hash functions naturally give rise to very "dense" parity constraints. To address this issue, we propose a technique to construct equivalent but sparser (and empirically easier to solve) parity constraints. Further, we introduce a more general version of WISH that relies *directly* on arbitrarily sparse parity constraints, thus giving rise to easier to solve MAP queries but providing weaker, one-sided guarantees on the approximation error for the partition function.

Our ILP formulation with sparsification techniques provides very good lower bounds on the partition function, while at the same time providing also upper bounds based on solving a sequence of LP relaxations. These upper bounds are much tighter than those obtained by tree decomposition and convexity [30]. This is a significant advance, because other state-of-the-art sampling based algorithms [10, 11, 13, 32] can usually provide probabilistic guarantees on lower bounds, but are not able to reason at all about upper bounds.

2 PROBLEM STATEMENT

We consider a discrete probabilistic graphical model [31] with $n = |V|$ random variables $\{x_i, i \in V\}$ where each random variable x_i takes values in a finite set $\mathcal{X}_i$. We consider a factor graph representation for a joint probability distribution over elements $x \in \mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$ (also referred to as **configurations**)

$$p(x) = \frac{1}{Z} \prod_{\alpha \in \mathcal{I}} \psi_\alpha(\{x\}_\alpha) \tag{1}$$

This is a compact representation for $p(x)$, which is defined as the product of non-negative factors $\psi_\alpha : \{x\}_\alpha \mapsto \mathbb{R}^+$, where $\mathcal{I}$ is an index set and $\{x\}_\alpha \subseteq V$ a subset of variables the factor ψ_α depends on. Z is a normalization constant known as *partition function* ensuring the probabilities sum up to one. Formally the partition function Z is defined as

$$Z = \sum_{x \in \mathcal{X}} \prod_{\alpha \in \mathcal{I}} \psi_\alpha(\{x\}_\alpha) = \sum_{x \in \mathcal{X}} w(x) \tag{2}$$

where for compactness we have introduced a weight function $w : \mathcal{X} \to \mathbb{R}^+$ that assigns to each configuration $x \in \mathcal{X}$ its unnormalized probability, namely

$$w(x) = \prod_{\alpha \in \mathcal{I}} \psi_\alpha(\{x\}_\alpha) \tag{3}$$

Computing the partition function Z is a #-P complete, intractable problem because it involves a sum over an exponentially large number of configurations. However, the partition function is a key property of a graphical model, needed e.g. to actually evaluate the probability of a configuration x under p. In this paper, we will focus on approximate techniques to estimate and bound this quantity. For simplicity, we consider the case of binary variables where $x_i \in \mathcal{X}_i = \{0, 1\}$. The general case can be encoded using a bit representation and binary variables.

3 BACKGROUND

This paper extends previous work by Ermon et al. [6] who introduced an algorithm called WISH to estimate the partition function (2). WISH is a randomized approximation algorithm that gives a constant factor approximation of Z with high probability. It involves solving a polynomial number of MAP inference queries for the graphical model conditioned on randomly generated evidence based on universal hashing.

3.1 FAMILIES OF HASH FUNCTIONS

A key ingredient of the WISH algorithm is the concept of **pairwise independent** hashing, originally introduced by Carter and Wegman [5] and later recognized as a tool that "should belong to the bag of tricks of every computer scientist" [33]. There are several in-depth expositions of the topic [cf. 12, 27, 28]. Here we will also make use of a weaker notion of hashing, called **uniform** hashing and defined as follows:

Definition 1. A family of functions $\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^m\}$ is called **uniform** if for $H \in_R \mathcal{H}$ it holds that $\forall x \in \{0,1\}^n$, the random variable $H(x)$ is uniformly distributed in $\{0,1\}^m$.

Here we use the notation $H \in_R \mathcal{H}$ to denote H being chosen uniformly at random from $\mathcal{H}$.

Definition 2. A family of functions $\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^m\}$ is called **pairwise independent** if it is uniform and for $H \in_R \mathcal{H}$ it holds that $\forall x_1, x_2 \in \{0,1\}^n$ with $x_1 \neq x_2$, the random variables $H(x_1)$ and $H(x_2)$ are independent.

Many constructions of pairwise independent hash functions are known. A simple and well-known one was used by Ermon et al.:

Proposition 1 ([6])**.** *Let* $A \in \{0,1\}^{m \times n}$, $b \in \{0,1\}^m$. *The family* $\mathcal{H}^{n,m} = \{h_{A,b}(x) : \{0,1\}^n \to \{0,1\}^m\}$ *where* $h_{A,b}(x) = Ax + b \mod 2$ *is a family of pairwise independent hash functions.*

Algorithm 1 WISH $(w, n = \log_2 |\mathcal{X}|, \delta, \alpha, \{\mathcal{H}^{n,i}\})$

```
$T \leftarrow \left\lceil \frac{\ln(1/\delta)}{\alpha} \ln n \right\rceil$
for $i = 0, \cdots, n$ do
    for $t = 1, \cdots, T$ do
        Sample hash function $h^i_{A,b}$ uniformly from $\mathcal{H}^{n,i}$
        $w^t_i \leftarrow \max_\sigma w(\sigma)$ subject to $h^i_{A,b}(\sigma) = \mathbf{0}$
    end for
    $M_i \leftarrow \text{Median}(w^1_i, \cdots, w^T_i)$
end for
Return $M_0 + \sum_{i=0}^{n-1} M_{i+1} 2^i$
```

3.2 THE WISH ALGORITHM FOR DISCRETE INTEGRATION

The basic idea behind WISH is to (implicitly) randomly partition the space of all possible configurations by universally hashing configurations into 2^m buckets. This step is achieved using randomly generated *parity constraints* of the form $Ax = b \mod 2$, which may also be viewed as logical XOR operations acting on the binary variables of the problem: $A_{i1}x_1 \oplus A_{i2}x_2 \oplus \cdots \oplus A_{in}x_n = b_i$. A combinatorial optimization solver is then used to find a configuration with the largest weight within a *single* bucket. This corresponds to solving a MAP query, i.e., solving an optimization problem subject to (randomly generated) parity constraints. By varying the number of buckets and repeating the process a small number of times, this strategy provably yields an estimate of the intractable normalization factor (2) within any desired degree of accuracy, with high probability and using only a polynomial number of MAP queries. For completeness, we provide the pseudocode for WISH as Algorithm 1, modified to have the hash families $\mathcal{H}^{n,i}, i \in \{0, 1, \ldots, n\}$, as parameters whose variations we will consider later[1]. We will write WISH($\{\mathcal{H}^{n,i}\}$) when the values of the other parameters are implicit.

Although MAP inference itself is an NP-hard problem, this strategy is still desirable considering that computing Z is a #P-hard problem, a complexity class believed to be even harder than NP. In practice, Ermon et al. [6] showed that the resulting MAP inference can be solved reasonably well using a state-of-the-art MAP inference engine called Toulbar [1], which was extended with custom propagators for parity constraints.

Theorem 1 ([6]). *For any $\delta > 0$, positive constant $\alpha \leq 0.0042$, and the hash families $\mathcal{H}^{n,i}$ given by Proposition 1, WISH($\{\mathcal{H}^{n,i}\}$) makes $\Theta(n \ln n \ln 1/\delta)$ MAP queries and, with probability at least $(1-\delta)$, outputs a 16-approximation of $Z = \sum_{\sigma \in \mathcal{X}} w(\sigma)$.*

[1]For $i = 0$, $h^i_{A,b} \equiv \mathbf{0}$ and no constraint is added.

Further, even if the MAP instances in the inner loop of Algorithm 1 are not solved to optimality, the output of the algorithm using suboptimal MAP solutions is an *approximate lower bound* for Z (specifically, no more than $16Z$) with probability at least $(1-\delta)$. If suboptimal solutions are within a constant factor L of the optimal, then the output is a $16L$-approximation of Z with probability at least $(1-\delta)$ [6]. Similarly, if one has access to upper bounds to the values of the MAP instances, the output of the algorithm using these upper bounds is an *approximate upper bound* (specifically, at least $1/16Z$) for Z with probability at least $(1-\delta)$.

3.3 IMPROVING WISH: HASHING USING TOEPLITZ MATRIX

The performance of Algorithm 1 can be improved by constructing pairwise independent hash functions not by choosing $A \in_R \{0,1\}^{i \times n}$ but rather letting A be a random $i \times n$ Toeplitz matrix [24]. Specifically, the first column and row of A are filled with uniform i.i.d. Bernoulli variables in $\{0,1\}$. The value of each entry is then copied into the corresponding descending top-left to bottom-right diagonal. This process requires $n + i - 1$ random bits rather than $ni = O(n^2)$. Let $\mathcal{T}(m,n) \subseteq \{0,1\}^{m \times n}$ be the set of $m \times n$ Toeplitz matrices with $0, 1$ entries. Then:

Proposition 2 ([12, 27]). *Let $A \in \mathcal{T}(m,n)$, $b \in \{0,1\}^m$. The family $\mathcal{H}^{n,m}_{\mathcal{T}} = \{h_{A,b}(x) : \{0,1\}^n \to \{0,1\}^m\}$ where $h_{A,b}(x) = Ax + b \mod 2$ is a family of pairwise independent hash functions.*

WISH($\{\mathcal{H}^{n,m}_{\mathcal{T}}\}$) still provides the same theoretical guarantees as Theorem 1 but has a more deterministic and stable behavior as it requires only $\Theta(n^2 \log n)$ random bits rather than $\Theta(n^3 \log n)$.

4 CONNECTIONS WITH CODING THEORY

For a problem with n binary variables, WISH requires solving $\Theta(n \log n)$ optimization instances. If these optimizations could be approximated (within a constant factor of the true optimal value) in polynomial time, this would give rise to a polynomial time algorithm that gives, with high probability, a constant factor approximation for the original counting problem. Note that this is a reasonable assumption, because perhaps the most interesting #-P complete counting problems are those whose corresponding decision problem are easy, e.g. counting weighted matchings in a graph (computing the permanent). A natural question arises: *are there interesting counting problems for which we can approximate* $\max_\sigma w(\sigma)$ *subject to* $A\sigma = b \mod 2$ *in polynomial time?*

To shed some light on this question, we show a connection with a decision problem arising in coding theory:

Definition 3 (MAXIMUM-LIKELIHOOD DECODING)**.** Given a binary $m \times n$ matrix A, a vector $b \in \{0,1\}^m$, and an integer $w > 0$, is there a vector $z \in \{0,1\}^n$ of Hamming weight $\leq w$, such that $Az = b \mod 2$?

As noted by Vardy [29], Berlekamp et al. [3] showed that this problem is NP-complete with a reduction from 3-DIMENSIONAL MATCHING. Further, Stern [26] and Arora et al. [2] proved that even approximating within any constant factor the solution to this problem is NP-hard.

These hardness results restrict the kind of problems we can hope to solve in our setting, which is more general. In fact, we can define a graphical model with single variable factors $\psi_i(x_i) = \exp(-x_i)$ for $x_i \in \{0,1\}$. Let $\mathcal{S} = \{x \in \{0,1\}^n : Ax = b \bmod 2\}$. Then

$$\max_{x\in\mathcal{S}} w(x) = \max_{x\in\mathcal{S}} \prod_{i=1}^{n} \psi_i(x_i) = \exp\left(\max_{x\in\mathcal{S}} \sum_{i=1}^{n} \log \psi_i(x_i)\right)$$
$$= \exp\left(\max_{x\in\mathcal{S}} -H(x)\right) = \exp\left(-\min_{x\in\mathcal{S}} H(x)\right)$$

where $H(x)$ is the Hamming weight of x. Thus, MAXIMUM-LIKELIHOOD DECODING of a binary code is a special case of MAP inference subject to parity constraints, but on a simple (disconnected) factor graph with factors acting only on single variable nodes. Intuitively, in the context of coding theory, there is a variable for each transmitted bit, and factors capture the probability of a transmission error on each bit. Thus there are no interactions between the variables, except for the ones introduced by the parity constraints $Ax = b \mod 2$, while in our context we allow for more complex probabilistic dependencies between variables specified as in Eq. (1). We therefore have the following theorem:

Theorem 2. *Given a binary $m \times n$ matrix A, a vector $b \in \{0,1\}^m$, and $w(x)$ as in Equation (3), the following optimization problem*

$$\max_{x\in\{0,1\}^n} \log w(x) \textit{ subject to } Ax = b \mod 2$$

is NP-hard to solve and to approximate within any constant factor.

Connections with coding theory is even deeper, and is not just an artifact of the particular hash function construction used. In fact, there is an intimate connection and a correspondence between universal hash functions and (binary) codes, where one can construct hash functions from binary codes and vice versa [27].

4.1 MESSAGE PASSING DECODING

Iterative Message Passing (MP) methods are among the most widely used decoding techniques. Although the decoding problem is computationally intractable, they usually have very good performance in practice [7, 19]. Since we can represent parity constraints as additional factors in our original factor graph model, MP techniques can also be heuristically applied to solve the more general MAP inference queries with parity constraints generated by WISH. Specifically, although a parity constraint over k variables would require a conditional probability table (CPT) of size 2^k to be specified, efficient Dynamic-Programming-based updates for parity constraints are known, see e.g. [19]. These updates have complexity which is linear in k, and thus, by representing parity constraints implicitly, we can directly use these techniques.

5 INTEGER PROGRAMMING FORMULATION

The NP-hard combinatorial optimization problem $\max_\sigma w(\sigma)$ subject to $A\sigma = b \mod 2$ can be formulated as an Integer Program [4]. This is a promising approach because Integer Linear Programs and related Linear programming (LP) relaxations have been shown to be a very effective at decoding binary codes by Feldman et al. [7]. Further, the empirically successful iterative message-passing decoding algorithms are closely related to LP relaxations of certain Integer Programs, either because they are directly trying to solve an LP or its dual like the MPLP and TRWBP [9, 25, 30], or attempting to approximately solve a variational problem over the same polytope like Loopy Belief Propagation [31].

5.1 MAP INFERENCE AS AN ILP

For simplicity, we consider the case of binary factors (pairwise interactions between variables), where equation (3) simplifies to $w(x) = \prod_{i\in V} \psi_i(x_i) \prod_{(i,j)\in E} \psi_{ij}(x_i, x_j)$ for some edge set E. Rewriting in terms of the logarithms, the unconstrained MAP inference problem can be stated as $\max_{x\in\{0,1\}^n} \sum_{i\in V} \theta_i(x_i) + \sum_{(i,j)\in E} \theta_{ij}(x_i, x_j)$ which can be written as an Integer Linear Program using binary indicator variables $\{\mu_i, i \in V\}$ and $\{\mu_{ij}(x_i, x_j), (i,j) \in E, x_i \in \{0,1\}, x_j \in \{0,1\}\}$ as follows [31]:

$$\max_{\mu_i, \mu_{ij}(x_i,x_j)} \sum_{i\in V} \theta_i(1)\mu_i + \theta_i(0)(1-\mu_i) +$$
$$\sum_{(i,j)\in E} \sum_{x_i,x_j} \theta_{ij}(x_i,x_j)\mu_{i,j}(x_i,x_j)$$

subject to

$$\forall i \in V, (i,j) \in E, \qquad \sum_{x_j \in \{0,1\}} \mu_{i,j}(0, x_j) = 1 - \mu_i$$

$$\forall i \in V, (i,j) \in E, \qquad \sum_{x_j \in \{0,1\}} \mu_{i,j}(1, x_j) = \mu_i$$

$$\forall i \in V, (i,j) \in E, \qquad \sum_{x_i \in \{0,1\}} \mu_{i,j}(x_i, 0) = 1 - \mu_j$$

$$\forall i \in V, (i,j) \in E, \qquad \sum_{x_i \in \{0,1\}} \mu_{i,j}(x_i, 1) = \mu_j$$

5.2 PARITY CONSTRAINTS

There are several possible encodings for the parity constraints $A\sigma = b \mod 2$, defining the so called **parity polytope** over $\sigma \in \mathbb{R}^n$. We summarize them next. Let $\mathcal{J}$ be the set of parity constraints (one entry per row of A). Let $\mathcal{N}(j)$ be the set of variables the j-th parity constraint depends on, namely the indexes of the non-zero columns of the j-th row of A[2]. We'll refer to $|\mathcal{N}(j)|$ as the length of the j-th XOR.

5.2.1 Exponential polytope representation

The simplest encoding is due to Jeroslow [16]. It requires that for all $j \in \mathcal{J}$, $S \subseteq \mathcal{N}(j)$, and $|S|$ odd, the following should hold

$$\sum_{i \in S} \mu_i + \sum_{i \in (\mathcal{N}(j) \setminus S)} (1 - \mu_i) \leq |\mathcal{N}(j)| - 1$$

Clearly, this requires a number of constraints that is exponential in the length of the XOR.

Another representation exponential in the length of the parity constraint is due to Feldman et al. [7]. For each S in the set $E_j = \{S \subseteq \mathcal{N}(j) : |S| \text{ even}\}$ there is an extra binary variable $w_{j,S} \in \{0,1\}$. It requires $\forall j \in \mathcal{J}, \sum_{S \in E_j} w_{j,S} = 1$ and $\forall j \in \mathcal{J}, \forall i \in \mathcal{N}(j), \mu_i = \sum_{S \in E_j : i \in S} w_{j,S}$.

5.2.2 Compact polytope representation

Yannakakis [34] introduced the following compact representation which requires only $O(n^3)$ variables and constraints, where n is the number of variables. For each constraint j, define $T_j = \{0, 2, \cdots, 2\lfloor|\mathcal{N}(j)|/2\rfloor\}$ as the set of even numbers between 0 and $|\mathcal{N}(j)|$.

- for all $j \in \mathcal{J}$ and for all $k \in T_j$ we have a binary variable $\alpha_{j,k} \in \{0,1\}$
- for all $j \in \mathcal{J}$ and for all $k \in T_j$ and for all $i \in \mathcal{N}(j)$ we have a binary variable $z_{i,j,k} \in \{0,1\}$, $0 \leq z_{i,j,k} \leq \alpha_{j,k}$

[2]To represent the desired parity of the j-th constraint imposed by b_j we use a dummy variable $d = 1$, and include d in $\mathcal{N}(j)$ whenever $b_j = 1$.

Then the following constraints are enforced:

$$\forall i \in V, j \in \mathcal{N}(i), \qquad \mu_i = \sum_{k \in T_j} z_{i,j,k}$$

$$\forall j \in \mathcal{J}, \qquad \sum_{k \in T_j} \alpha_{j,k} = 1$$

$$\forall j \in \mathcal{J}, \forall k \in T_j, \qquad \sum_{i \in \mathcal{N}(j)} z_{i,j,k} = k\alpha_{j,k}$$

For any set of parity constraints, these 3 encodings are equivalent, in the sense that the subset of $\{\mu_i\}$ satisfying the constraints is the same [7]. Thus, the corresponding MAP inference problems are also equivalent, as the objective function, by expressing each $\mu_{i,j}$ in terms of the $\{\mu_i\}$ variables, can be re-written as a (possibly non-linear) function of only $\{\mu_i\}$.

5.3 SOLVING INTEGER PROGRAMS

Solving ILPs typically relies on solving a sequence of Linear Programming (LP) relaxations obtained by relaxing the integrality constraints. The solution to the relaxation provides an upper bound to the original integer maximization problem. Since LP can be solved in polynomial time, using Theorem 1 and following remarks we have a **polynomial time** method to obtain approximate upper bounds on the partition function which hold with high probability, although without tightness guarantees. Notice that upper bounds of this form could also be obtained using message passing techniques such as MPLP or TRWBP [9, 25, 30], which can also provide upper bounds to the values of the MAP inference queries in the inner loop of WISH.

IP solvers such as IBM ILOG CPLEX Optimization Studio [15] solve a sequence of LP relaxations based on branching on the problems's variables, iteratively improving the upper bound and keeping track of the best integer solution found, until lower and upper bounds match. Thus, one advantage of using an IP solver over standard Message Passing techniques is that the upper and lower bounds improve over time, and it is guaranteed to eventually provide an optimal solution for the original integer problem. In Figure 1 we plot the upper bound reported by CPLEX as a function of runtime for a random 10×10 Ising model with mixed interactions. It's clear that there is quickly a dramatic improvement over the value of the basic LP relaxation, which is the value reported by CPLEX around time zero, and that the upper bound keeps improving although at a slower rate. We note that other techniques such as by Sontag et al. [25] could also be used to iteratively tighten the LP relaxation, and might lead to better scaling behavior on certain classes of very large problems [35].

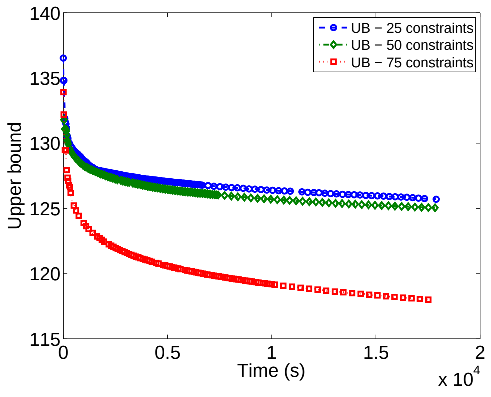


Figure 1: Upper bound as a function of runtime.

5.4 INDUCING SPARSITY

As we have shown, solving MAP inference queries subject to parity constraints is hard in general. However, adding parity constraints can sometimes makes the optimization easier. For example, when A is the identity matrix, enforcing $A\sigma = b \mod 2$ corresponds to fixing the values of all variables and leads to a trivial optimization problem. Empirically, **sparse** constraints, such as the ones used in low density parity check (LDPC) codes from Gallager [8], tend to be much easier to solve. Unfortunately, constructions in both Propositions 1 and 2 to create pairwise independent hash functions require parity constraints that are of average length $n/2$, i.e., the corresponding matrix A is not sparse.

A set of parity constraints specified through matrices A, b defines a set of solutions $\mathcal{S} = \{x \in \{0,1\}^n : Ax = b\}$, which is the translated nullspace of the matrix A. The nullspace is a vector space, defined with operations over the finite field $\mathbb{F}(2)$, i.e. modular arithmetic. Exploiting basic linear algebraic properties, it can be shown that applying **elementary row operations** to $[A|b]$ does not change the solution set $\mathcal{S}$ and thus the optimization problem. On the other hand, the parity polytope we described earlier is *not* a function of the solution set $\mathcal{S}$ but *depends explicitly* on the form of the matrices A and b. This fact was also noted by Feldman et al. [7], who showed that a new matrix $[A'|b']$ constructed from $[A|b]$ by adding new rows that are linear combinations of the rows of $[A|b]$ can lead to a tighter LP relaxation, although $Ax = b$ and $A'x = b'$ define the same solution set $\mathcal{S}$ (because the constraints added are all implied).

In this paper we propose to exploit these facts and rewrite the constraints in a form that is equivalent, i.e., defines the same set of solutions, but is easier to solve. Specifically, given a a set of parity constraints specified through matrices A, b we look for matrices A', b' that define the same set of solutions, namely $\{x \in \{0,1\}^n : Ax = b\} = \{x \in \{0,1\}^n : A'x = b'\}$ but are much sparser, namely $||[A'|b']||_1 \ll ||[A|b]||_1$. Unfortunately, even finding a sparse linear combination of the rows is computationally intractable, as it can be seen as an instance of MAXIMUM-LIKELIHOOD DECODING, where the code is given in terms of the generators (the rows of A) rather than the check matrix. We therefore propose to use two approaches:

- Perform Gauss-Jordan elimination on $[A|b]$ to convert $[A|b]$ to **reduced row echelon form**;
- Try all combinations of up to k rows $r_1, \cdots, r_k$ of $[A|b]$, and if their sum $r_1 \oplus \cdots \oplus r_k$ is sparser than any of the r_i, substitute r_i with $r_1 \oplus \cdots \oplus r_k$.

Both techniques are based on **elementary row operations** and therefore are guaranteed to maintain the solution set $\mathcal{S}$ and to improve sparsity.

In Figure 2 we show the median upper and lower bounds found by CPLEX for several randomly generated constraints on a random 10×10 Ising grid model with mixed interactions. Starting with a matrix A generated using the Toeplitz matrix construction in Proposition 2, we run CPLEX for 10 minutes with and without sparsification, reporting the best upper and lower bounds found. We see that without any preprocessing (NoPre) CPLEX fails at finding any integer solution when there are more than 15 parity constraints. Performing Gauss-Jordan elimination (Diag) significantly improves both the upper bound and the lower bound. The effect is particularly significant for a large number of constraints, when the reduced row echelon form of A is close to the identity matrix. Adding the additional greedy substitution step (DiagGreedy, looking at all combinations of up to $k = 4$ rows) slightly improves the quality of the upper bound, but the lower bound significantly degrades. Therefore, for the rest of the paper we will use only Gauss-Jordan elimination preprocessing.

6 LOWER BOUNDS: SHORT XORS

As mentioned in Section 3.2, one practical way to obtain lower bounds from the WISH algorithm is to use suboptimal solutions of the underlying MAP inference problems. Here we explore a different way, namely, using sparse or *short* parity constraints (XORs), which are often easier for constraint solvers to reason about. This results in a family of hash functions that is uniform *but not pairwise independent*, leading to a weaker but practically valuable version of Theorem 1.

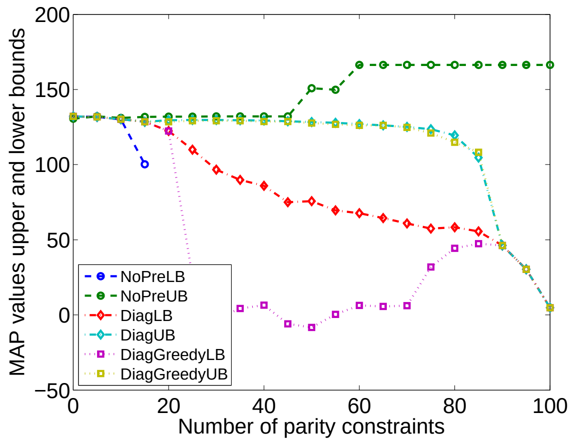


Figure 2: Upper and lower bounds with and without sparsification.

6.1 WISH WITH UNIFORM HASHING

Fix *any* subset $\mathcal{A}^{m\times n} \subseteq \{0,1\}^{m\times n}$ of $m \times n$ matrices. We will later create short XORs by choosing $\mathcal{A}^{m\times n}$ such that every row of every matrix in this set has only $k \ll n/2$ non-zero entries.

Proposition 3. *Let $A \in \mathcal{A}^{m\times n}$ and $b \in \{0,1\}^m$. The family $\mathcal{H}^{n,m} = \{h_{A,b}(x) : \{0,1\}^n \to \{0,1\}^m\}$ where $h_{A,b}(x) = Ax + b \mod 2$ is a family of* ***uniform*** *hash functions.*

Proof. Let $x \in \{0,1\}^n$, $A \in_R \mathcal{A}^{m\times n}$, $b \in_R \{0,1\}^m$, and a_i denote the i-th row of A. Then, for all i:

$$\begin{aligned}\Pr[a_i x \oplus b_i = 0] &= \sum_{v\in\{0,1\}^n} \Pr[a_i = v]\Pr[vx \oplus b_i = 0] \\ &= \sum_{v\in\{0,1\}^n} \Pr[a_i = v]\Pr[b_i = vx \bmod 2] \\ &= \sum_{v\in\{0,1\}^n} \Pr[a_i = v]\ \frac{1}{2}\ = \frac{1}{2}\end{aligned}$$

Hence, $\Pr[Ax+b = 0 \bmod 2] = \prod_i \Pr[a_i x + b_i = 0 \bmod 2] = \frac{1}{2^m}$ for any x, proving uniformity. □

With such a family of hash functions, Theorem 1 as such does not hold, but the following weaker, one-directional version still does:

Theorem 3. *For any $\delta > 0$, positive constant $\alpha \le 0.0042$, and families $\mathcal{H}^{n,i}$ of uniform (but not necessarily pairwise independent) hash functions, with probability at least $(1-\delta)$, WISH($\{\mathcal{H}^{n,i}\}$) outputs an estimate no larger than $16Z = 16\sum_{\sigma\in\mathcal{X}} w(\sigma)$.*

In other words, even without pairwise independence, the output divided by 16 is a lower bound with high probability. To prove this result, we employ a proof strategy similar to the one used by Ermon et al. [6]. For completeness, we start by stating some definitions we borrow from that work:

Definition 4 ([6]). Fix an ordering $\sigma_i, 1 \le i \le 2^n$, of the configurations in $\mathcal{X}$ such that for $1 \le j < 2^n$, $w(\sigma_j) \ge w(\sigma_{j+1})$. For $i \in \{0, 1, \cdots, n\}$, define $b_i \triangleq w(\sigma_{2^i})$. Define a special *bin* $B \triangleq \{\sigma_1\}$ and, for $i \in \{0, 1, \cdots, n-1\}$, define *bin* $B_i \triangleq \{\sigma_{2^i+1}, \sigma_{2^i+2}, \cdots, \sigma_{2^{i+1}}\}$.

Next we prove a new bound on M_i that holds **regardless of pairwise independence**:

Lemma 1. *Suppose $h^i_{A,b}$ is chosen from a family $\mathcal{H}^{n,i}$ of universal (but not necessarily pairwise independent) hash functions. Let $M_i = \mathrm{Median}(w^1_i, \cdots, w^T_i)$ be defined as in Algorithm 1 and b_i as in Definition 4. Then, for all $c \ge 2$, there exists an $\alpha^*(c) > 0$ such that for $0 < \alpha \le \alpha^*(c)$,*

$$\Pr\left[M_i \le b_{\max\{i-c,0\}}\right] \ge 1 - \exp(-\alpha T)$$

Proof. The statement trivially holds when $i - c \le 0$. Otherwise, let us define the set of the 2^j heaviest configurations as in Definition 4, $\mathcal{X}_j = \{\sigma_1, \sigma_2, \cdots, \sigma_{2^j}\}$. Define the following random variable $S_j(h^i_{A,b}) \triangleq \sum_{\sigma\in\mathcal{X}_j} 1_{\{A\sigma = b \bmod 2\}}$ which gives the number of elements of $\mathcal{X}_j$ satisfying the random parity constraints $A\sigma = b \bmod 2$. The randomness is over the choice of A and b when $h^i_{A,b}$ is sampled from $\mathcal{H}^{n,i}$. Since $\mathcal{H}^{n,i}$ is a family of uniform hash functions, by definition for any σ the random variable $1_{\{A\sigma=b \bmod 2\}}$ is Bernoulli distributed with probability $1/2^i$. Then it follows that $\mathbb{E}[S_j(h^i_{A,b})] = \sum_{\sigma\in\mathcal{X}_j} 1/2^i = \frac{|\mathcal{X}_j|}{2^i} = 2^{j-i}$.

The random variable w_i is defined as $w_i = \max_\sigma w(\sigma)$ subject to $A\sigma = b \mod 2$. Then we have:

$$\Pr[w_i \le b_j] = \Pr[w_i \le w(\sigma_{2^j})] \ge \Pr[S_j(h^i_{A,b}) < 1]$$

which is the probability that no configuration from $\mathcal{X}_j$ satisfies i randomly chosen parity constraints. Notice that $S_j(h^i_{A,b})$ is non-negative, hence from Markov's Inequality, $\Pr[S_j(h^i_{A,b}) \ge 1] \le \mathbb{E}[S_j(h^i_{A,b})] = 2^{j-i}$. Thus for $j = i - c$ and $c \ge 2$, we have:

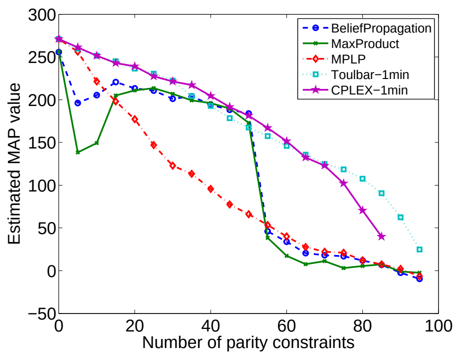
$$\Pr[w_i \le b_{i-c}] \ge \Pr[S_{i-c}(h^i_{A,b}) < 1] \ge 1 - 2^{-c} \ge 3/4$$

Finally, since $w^1_i, \cdots, w^T_i$ are i.i.d. realizations of w_i, we can apply Chernoff's Inequality to the corresponding indicator variables $I_t = I(w^t_i \le b_{i-c})$ each with mean $\ge 3/4$ and obtain:

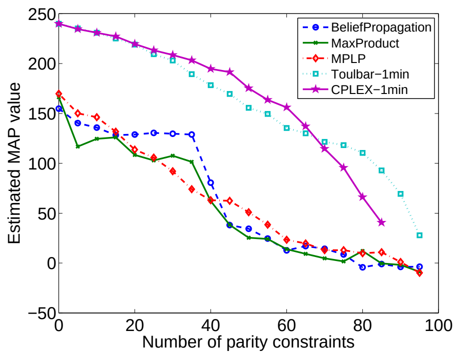
$$\Pr\left[M_i \le b_{i-c}\right] = \Pr\left[\sum_t I_t \ge T/2\right] \ge 1 - \exp(-\alpha^*(c)T)$$

where $\alpha^*(2) = 2(3/4 - 1/2)^2 = 1/8$. □

With this new lemma, we have all we need to prove Theorem 3. The proof is similar to the one of Theorem 1 and is not included for space reasons.



(a) Attractive 10×10. Length 4 Xors



(b) Mixed 10×10. Length 4 Xors

Figure 3: Optimization with short parity constraints.

6.2 EVALUATION OF SHORT XORS

As briefly alluded to earlier, we will use *short XORs* in order to make MAP inference more efficient in practice. Specifically, we will choose A uniformly from $\mathcal{A}_k^{m\times n} \subseteq \{0,1\}^{m\times n}$, which is the set of matrices such that every row has only k non-zero entries, where k will typically be much smaller than $n/2$. In general, smaller values of k lead to faster execution of WISH but at the cost of weaker lower bounds.

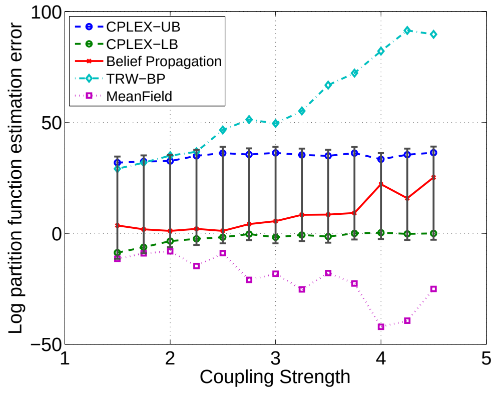
Figure 3 compares several approaches to solve the MAP inference problems constrained by random parity constraints for a 10×10 Ising grid model with attractive and mixed interactions (external field $f = 1.0$ and weight $w = 3.0$; see below for a formal description of the probabilistic model used). We compare three message passing approaches, namely Belief Propagation (BP), Max-Product (MP), and MPLP [9], and two combinatorial optimization solvers, namely Toulbar [1] and CPLEX 12.3 [15], both with a 1 minute time limit. We show the median value of the solution found over 50 realizations, for each number of parity constraints added. We run the Message Passing methods until they find a feasible solution satisfying the parity constraints or up to 10000 iterations. If no feasible solution is found, we round the final beliefs to an integer solution and project it on the feasible set by solving the linear equations with Gaussian Elimination, thus changing the value of some of the variables. For this problem, using "long" parity constraints of length 50, Message Passing methods can only find feasible solutions for up to 10 constraints (consistent with CPLEX performance in Figure 2). In contrast, as shown in Figure 3, using short XORs of length 4 (typical values encountered e.g. for low density parity check codes), Message Passing methods can find feasible solutions for up to about 40−50 constraints, at which point there is a significant performance drop caused by the need for a projection step. We see that for the attractive case, Message Passing methods are competitive with combinatorial optimization approaches but only for a moderate number of constraints. In the more challenging mixed interactions case, CPLEX and Toulbar appear to be clearly superior. We think the the unsatisfactory performance of message passing techniques (compared e.g. to when used for LDPC decoding) is caused by the more complicated probabilistic dependencies imposed by the Ising model, which is much more intricate than a typical transmission error model.

7 EXPERIMENTS

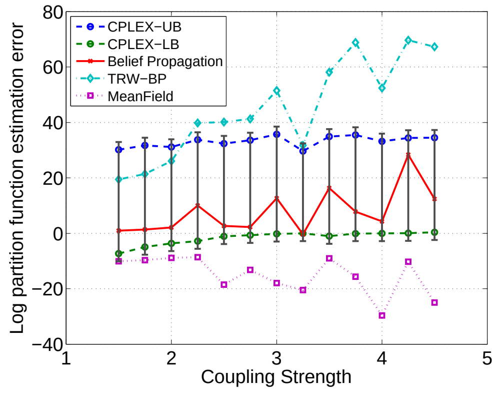
We evaluate the performance of WISH augmented with Toeplitz-matrix based hash functions (from Proposition 2) and CPLEX 12.3 [15] to solve the ILP formulation of the MAP queries. All the optimization instances are solved in parallel on a compute cluster, with a timeout of 10 minutes on Intel Xeon 5670 3GHz machines.We use Gauss-Jordan elimination preprocessing to improve the quality of the LP relaxations. We use the Jaroslow encoding for parity constraints $j \in \mathcal{J}$ such that $|N(j)| \le 10$, and the Yannakakis encoding otherwise. We evaluate the lower bound and upper bounds for the partition functions of $M \times M$ grid Ising models for $M \in \{10, 15\}$, with random interactions (positive and negative) and external field $f \in \{0.1, 1.0\}$. Specifically, there are M^2 binary variables, with single node potentials $\psi_i(x_i) = \exp(f_i x_i)$ and pairwise interactions $\psi_{ij}(x_i, x_j) = \exp(w_{ij} x_i x_j)$, where $w_{ij} \in_R [-w, w]$ and $f_i \in_R [-f, f]$.

We compare with Loopy BP [23] which estimates Z, Tree Reweighted BP [30] which gives a provable upper bound, and the Mean Field approach [31] which gives a provable lower bound. We use the implementations in the LibDAI library [22] and compare with ground truth obtained using the Junction Tree method [20].

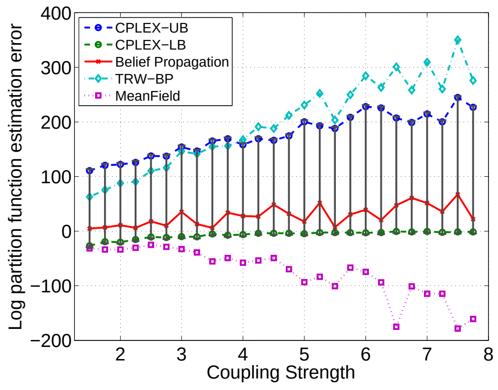
Figure 4 shows the error in the resulting estimates, together with the upper and lower bounds obtained with WISH augmented with Toeplitz-matrix hashing and CPLEX. We immediately see that our *lower bounds*



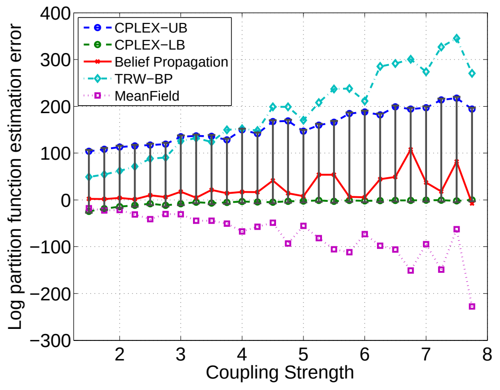
(a) Mixed 10 × 10. Field 0.1.



(b) Mixed 10 × 10. Field 1.0.



(c) Mixed 15 × 15. Field 0.1.



(d) Mixed 15 × 15. Field 1.0.

Figure 4: Results on spin glasses grids.

are highly accurate (error close to 0), which means that the lower bounds provided by CPLEX for the ILPs must be close to optimality. Similarly good lower bounds can also be obtained using the original WISH algorithm [6], and also using SampleSearch [11]. However, neither SampleSearch nor the original WISH (without LP relaxations) provide *upper bound* guarantees, only the TRWBP approach does. Specifically, the original WISH algorithm with Toulbar [1] provides an upper bound only upon proving optimality for all optimization instances in the inner loop. In contrast, the ILP formulation provides us with **anytime** and gradually improving upper bounds based on LP relaxations (cf. Figure 1), often well before it can actually solve the problems to optimality (which might not be possible on larger instances) or, in principle, even before it can find a feasible solution. Figure 4 shows that our upper bounds are significantly tighter than the ones obtained using TRWBP in the hard weights region. Further, our ILP approach is guaranteed to eventually give an accurate answer, within a constant factor, given enough time. In contrast, message passing techniques are usually quite fast (if they converge) but do not provide better results with more runtime.

8 CONCLUSIONS

We explored several extensions of the recent WISH [6] algorithm for computing discrete integrals. First, we used a better, more deterministic and thus more stable construction for pairwise independent hash functions. Using a connection with max-likelihood decoding of binary codes, we showed that the MAP inference queries generated by WISH are in general not polynomial time solvable or even approximable. On the positive side, this led to the use of an ILP formulation for the problem, inspired by iterative message passing decoding. To increase the practicality of the ILP approach, we sparsified parity constraints while preserving their desirable properties. Further, we extended WISH to directly utilize uniform but not necessarily pairwise independent hash functions, leading to computationally easier optimization problems while still providing probabilistic lower bound guarantees. Finally, we showed that by solving a sequence of LP relaxations we can obtain not only very accurate lower bounds but also upper bounds that are much tighter than the ones provided by TRWBP, which is based on tree decomposition and convexity.

References

[1] D. Allouche, S. de Givry, and T. Schiex. Toulbar2, an open source exact cost function network solver. Technical report, INRIA, 2010.

[2] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. In *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*, pp. 724–733. IEEE, 1993.

[3] E. Berlekamp, R. McEliece, and H. Van Tilborg. On the inherent intractability of certain coding problems. *Information Theory, IEEE Transactions on*, 24(3): 384–386, 1978.

[4] D. Bertsimas and J. N. Tsitsiklis. *Introduction to linear optimization*. Athena Scientific Belmont, MA, 1997.

[5] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of computer and system sciences*, 18(2):143–154, 1979.

[6] S. Ermon, C. Gomes, A. Sabharwal, and B. Selman. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *ICML (To appear)*, 2013.

[7] J. Feldman, M. J. Wainwright, and D. R. Karger. Using linear programming to decode binary linear codes. *Information Theory, IEEE Transactions on*, 51(3): 954–972, 2005.

[8] R. Gallager. Low-density parity-check codes. *Information Theory, IRE Transactions on*, 8(1):21–28, 1962.

[9] 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.

[10] V. Gogate and R. Dechter. SampleSearch: A scheme that searches for consistent samples. In *Proc. 10th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2007.

[11] V. Gogate and R. Dechter. SampleSearch: Importance sampling in presence of determinism. *Artificial Intelligence*, 175(2):694–729, 2011.

[12] O. Goldreich. Randomized methods in computation. *Lecture Notes*, 2011.

[13] C. Gomes, A. Sabharwal, and B. Selman. Model counting: A new strategy for obtaining good bounds. In *AAAI*, pp. 54–61, 2006.

[14] C. Gomes, A. Sabharwal, and B. Selman. Near-uniform sampling of combinatorial spaces using XOR constraints. *Advances In Neural Information Processing Systems*, 19:481–488, 2006.

[15] IBM ILOG. IBM ILOG CPLEX Optimization Studio 12.3, 2011.

[16] R. Jeroslow. On defining sets of vertices of the hypercube by linear inequalities. *Discrete Mathematics*, 11 (2):119–124, 1975.

[17] M. Jerrum and A. Sinclair. The Markov chain Monte Carlo method: an approach to approximate counting and integration. *Approximation algorithms for NP-hard problems*, pp. 482–520, 1997.

[18] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[19] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519, 2001.

[20] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 157–224, 1988.

[21] N. Madras. *Lectures on Monte Carlo Methods*. American Mathematical Society, 2002. ISBN 0821829785.

[22] J. Mooij. libDAI: A free and open source c++ library for discrete approximate inference in graphical models. *JMLR*, 11:2169–2173, 2010.

[23] K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI*, 1999.

[24] K. B. Petersen and M. S. Pedersen. The matrix cookbook. *Technical University of Denmark*, pp. 7–15, 2008.

[25] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *UAI*, 2008.

[26] J. Stern. Approximating the number of error locations within a constant ratio is np-complete. In *Proceedings of the 10th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pp. 325–331. Springer-Verlag, 1993.

[27] D. R. Stinson. On the connections between universal hashing, combinatorial designs and error-correcting codes. *Congressus Numerantium*, pp. 7–28, 1996.

[28] S. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 2011.

[29] A. Vardy. Algorithmic complexity in coding theory and the minimum distance problem. In *STOC*, 1997.

[30] M. Wainwright. Tree-reweighted belief propagation algorithms and approximate ML estimation via pseudo-moment matching. In *AISTATS*, 2003.

[31] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

[32] W. Wei and B. Selman. A new approach to model counting. In *Theory and Applications of Satisfiability Testing (SAT)*, pp. 324–339, 2005.

[33] A. Wigderson. Lectures on the fusion method and derandomization. Technical report, Technical Report SOCS-95.2, School of Computer Science, McGill University, 1995.

[34] M. Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43(3):441–466, 1991.

[35] C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation–an empirical study. *The Journal of Machine Learning Research*, 7:1887–1907, 2006.

Monte-Carlo Planning: Theoretically Fast Convergence Meets Practical Efficiency

Zohar Feldman and **Carmel Domshlak**
Faculty of Industrial Engineering and Management
Technion, Israel

Abstract

Popular Monte-Carlo tree search (MCTS) algorithms for online planning, such as ε-greedy tree search and UCT, aim at rapidly identifying a reasonably good action, but provide rather poor worst-case guarantees on performance improvement over time. In contrast, a recently introduced MCTS algorithm BRUE guarantees exponential-rate improvement over time, yet it is not geared towards identifying reasonably good choices right at the go. We take a stand on the individual strengths of these two classes of algorithms, and show how they can be effectively connected. We then rationalize a principle of "selective tree expansion", and suggest a concrete implementation of this principle within MCTS. The resulting algorithms favorably compete with other MCTS algorithms under short planning times, while preserving the attractive convergence properties of BRUE.

1 INTRODUCTION

Markov decision processes (MDPs) are a standard model for planning under uncertainty [19]. An MDP $\langle S, A, Tr, R\rangle$ is defined by a set of possible agent states S, a set of agent actions A, a stochastic transition function $Tr : S \times A \times S \to [0, 1]$, and a reward function $R : S \times A \times S \to \mathbb{R}$. The current state of the agent is fully observable, and the objective of the agent is to act so to maximize its accumulated reward. In the finite horizon setting considered here, the reward is accumulated over some predefined number of steps H. The description of the MDP is assumed to be concise, and, depending on the problem domain and the representation language, it can be either declarative or generative (or mixed). While declarative models provide the agents with greater algorithmic flexibility, generative models are more expressive, and both types of models allow for simulated execution of all feasible action sequences, from any state of the MDP.

In online planning for MDPs, the agent focuses on its current state only, deliberates about the set of possible policies from that state onwards and, when interrupted, uses the outcome of that exploratory deliberation to choose what action to perform next. The quality of the action a, chosen for state s with H steps-to-go, is assessed in terms of the probability that a is sub-optimal, or in terms of the (closely related) measure of simple regret. The latter captures the performance loss that results from taking a and then following an optimal policy π^* for the remaining $H-1$ steps, instead of following π^* from the beginning [5].

With a few recent exceptions developed for declarative MDPs [4, 16, 7], most algorithms for online MDP planning constitute variants of what is called Monte-Carlo tree search (MCTS) [23, 18, 15, 9, 8, 21, 25]. Most MCTS algorithms for online planning, such as ε-greedy tree search and UCT, aim at rapidly identifying a reasonably good action, but offer only polynomial-rate reduction of simple regret over the deliberation time. In contrast, a recently introduced MCTS algorithm BRUE guarantees exponential-rate reduction of simple regret over time, yet it does not make special efforts to home in on a reasonable alternative fast [11]. Of course, "good" is often the best one can hope for in large MDPs of interest under practically reasonable deliberation-time allowances. Therefore, as we reconfirm by an evaluation on benchmarks from the recent probabilistic planning competition, BRUE is often empirically inferior to its (guarantees-wise inferior) competitors.

Reflecting on the differences between the two types of algorithms, here we show that BRUE can be redesigned to perform extremely well also under early interruptions of planning, and this without compromising much neither the long-term empirical performance nor theoretical guarantees. We do that in two steps. First, we connect between the iterative tree expansion of the standard MCTS scheme and the "separation of concerns" principle that underlies BRUE. The resulting modification of BRUE, $\mathsf{BRUE}_{\mathcal{I}}$, already substantially improves over BRUE in short-term effectiveness. Building upon $\mathsf{BRUE}_{\mathcal{I}}$, we then introduce a machinery of *selective tree expansion* that further pushes the boundaries of online MDP planning. Viewing MCTS as a message passing within the hierarchy of forecasters, this mechanism is based on (i) classifying the roles that different forecasters in the hierarchy should fulfill in order to improve the quality of the decision at the root, and on (ii) exploit this classification to adaptively decide *how* each forecaster should aim at fulfilling its role. As testified by our empirical evaluation, the resulting algorithm, $\mathsf{BRUE}_{\mathcal{IC}}$, favorably and robustly competes with other MCTS algorithms under short planning times, while preserving both the attractive formal properties of BRUE, as well as the empirical strength of the latter under permissive deliberation-time allowances.

2 BACKGROUND

Henceforth, $\mathbf{\Pi}$ denotes the set of all valid policies for the MDP in question, $A(s) \subseteq A$ denotes the actions applicable in state s, the operation of drawing a sample from a distribution $\mathcal{D}$ over set $\aleph$ is denoted by $\sim \mathcal{D}[\aleph]$, $\mathcal{U}$ denotes uniform distribution, and $[\![n]\!]$ for $n \in \mathbb{N}$ denotes the set $\{1, \ldots, n\}$.

2.1 CANONICAL MCTS SCHEME

MCTS, a canonical scheme underlying various MCTS algorithms for online MDP planning, is depicted in Figure 1a. Starting with the current state s_0, MCTS performs an iterative construction of a tree[1] $\mathcal{T}$ rooted at s_0. At each iteration, MCTS rollouts a state-space sample ρ from s_0, which is then used to update $\mathcal{T}$. First, each state/action pair (s, a) is associated with a counter $n(s, a)$ and a value accumulator $\widehat{Q}(s, a)$, both initialized to 0. When a sample ρ is rolled out, for all states $s_i \in \rho \cap \mathcal{T}$, $n(s_i, a_{i+1})$ and $\widehat{Q}(s_i, a_{i+1})$ are updated on the basis of ρ by the UPDATE-NODE procedure. Second, $\mathcal{T}$ can also be expanded with any part of ρ; The standard choice is to expand $\mathcal{T}$ with only the first state along ρ that is new to $\mathcal{T}$. In any case, once the sampling is interrupted, MCTS uses the information stored at the tree's root to recommend an action to perform in s_0.

[1]In MDPs, there is no reason to distinguish between nodes associated with the same state at the same depth. Hence, the graph $\mathcal{T}$ constructed by MCTS instances typically forms a DAG. Nevertheless, for consistency with prior literature, we stay with the term "tree".

```
MCTS: [input: $\langle S, A, Tr, R\rangle$; $s_0 \in S$]
  search tree $\mathcal{T} \leftarrow$ root node $s_0$
  for $n \leftarrow 1 \ldots$ time permits do
      PROBE$(s_0, 0)$
  return $\arg\max_a \widehat{Q}(s_0, a)$
- - - - - - - - - - - - - - - - - - - - - - - - - -
PROBE $(s$ : state, $d$ : depth$)$
  if END-OF-PROBE$(s, d)$ then return EVALUATE$(s, d)$
  $a \leftarrow$ ROLLOUT-POLICY$(s)$
  $s' \sim P(S \mid s, a)$
  $r \leftarrow R(s, a, s') +$ PROBE$(s', d+1)$
  UPDATE-NODE$(s, a, r)$
  return $r$
                              (a)
```

```
END-OF-PROBE $(s$ : state, $d$ : depth$)$
  if $s \notin \mathcal{T}$ then
    add $s$ to $\mathcal{T}$ and return true
  else if $d = H$ then return true else return false
- - - - - - - - - - - - - - - - - - - - - - - - - -
EVALUATE $(s$ : state, $d$ : depth$)$
  for $t \leftarrow d \ldots H$ do
      $a \sim \mathcal{U}[A(s)]$
      $s' \sim P(S \mid s, a)$
      $r \leftarrow r + R(s, a, s')$
      $s \leftarrow s'$
  return $r$
- - - - - - - - - - - - - - - - - - - - - - - - - -
UPDATE-NODE $(s$ : state, $a$: action, $r$ : reward$)$
  $n(s, a) \leftarrow n(s, a) + 1$
  $\widehat{Q}(s, a) \leftarrow \widehat{Q}(s, a) + \frac{r - \widehat{Q}(s,a)}{n(s,a)}$
- - - - - - - - - - - - - - - - - - - - - - - - - -
ROLLOUT-POLICY $(s$ : state$)$
  if $n(s, a) = 0$ for some $a \in A(s)$ then
    return $a \sim \mathcal{U}[\{a \in A(s) \mid n(s, a) = 0\}]$
  else
    $n(s) \leftarrow \sum_{a \in A(s)} n(s, a)$
    return $\text{argmax}_a \left[\widehat{Q}(s, a) + c\sqrt{\frac{\log n(s)}{n(s,a)}}\right]$
                              (b)
```

Figure 1: (a) Monte-Carlo tree search template, and (b) the UCT specifics.

Numerous concrete instances of MCTS have been proposed, with ε-greedy [23] probably being the most widely known, and UCT [15] and its modifications [9, 24] being the most popular such instances these days [12, 22, 3, 2, 10, 14]. Concrete instances of MCTS vary mostly along the implementation of the ROLLOUT-POLICY sub-routine, that is, in their policies for directing the rollout within $\mathcal{T}$. For instance, the specific ROLLOUT-POLICY of UCT is shown in Figure 1b. This policy is based on the deterministic decision rule UCB1 [1], originally proposed for optimal balance between exploration and exploitation for cumulative regret minimization in stochastic multi-armed bandit (MAB) problems [20]. In general, different instances of MCTS vary in their balance between exploration and exploitation. However, it has already been noticed that exploitation may considerably slow down the reduction of simple regret over time [6]. In-

```
MCTS2e: [input: $\langle S, A, Tr, R\rangle$; $s_0 \in S$]
  search tree $\mathcal{T} \leftarrow$ root node $s_0$; $\sigma \leftarrow 0$
  for $n \leftarrow 1 \ldots$ time permits do
      $\sigma \leftarrow$ SWITCH-FUNCTION$(n, \sigma)$
      PROBE$(s_0, 0, \sigma)$
  return $\arg\max_a \widehat{Q}(s_0, a)$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
PROBE $(s$ : state, $d$ : depth, $\sigma \in [\![H]\!])$
  if END-OF-PROBE$(s, d)$ then return EVALUATE$(s, d)$
  if $d < \sigma$ then
      $a \leftarrow$ EXPLORATION-POLICY$(s)$
  else
      $a \leftarrow$ ESTIMATION-POLICY$(s)$
  $s' \sim P(S \mid s, a)$
  $r \leftarrow R(s, a, s') +$ PROBE$(s', d+1, \sigma)$
  if $d = \sigma$ then UPDATE-NODE$(s, a, r)$
  return $r$
```

(a)

```
END-OF-PROBE $(s$ : state, $d$ : depth)
  if $d = H$ then return true else return false
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
EVALUATE $(s$ : state, $d$ : depth)
  return 0
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
UPDATE-NODE $(s$ : state, $a$: action, $r$ : reward)
  if $s \notin \mathcal{T}$ then add $s$ to $\mathcal{T}$
  $n(s, a) \leftarrow n(s, a) + 1$
  $\widehat{Q}(s, a) \leftarrow \widehat{Q}(s, a) + \frac{r - \widehat{Q}(s,a)}{n(s,a)}$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
SWITCH-FUNCTION $(n$ : iteration, $\sigma \in [\![H]\!])$
  return $H - ((n-1) \mod H)$ // round robin on $[\![H]\!]$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
EXPLORATION-POLICY $(s$ : state)
  return $a \sim \mathcal{U}[A(s)]$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
ESTIMATION-POLICY $(s$ : state)
  return $a \sim \mathcal{U}\left[\{a \mid \arg\max_{a \in A(s)} \widehat{Q}(s, a)\}\right]$
```

(b)

Figure 2: Monte-Carlo tree search with "separation of concerns" (a), and the BRUE specifics (b).

deed, UCB1 (and thus UCT) achieves only polynomial-rate reduction of simple regret over time [6], and the number of samples after which the bounds of UCT on simple regret become meaningful might be as high as hyper-exponential in H [9]. In fact, no instance of the MCTS scheme (Figure 1a) suggested so far breaks the barrier of the worst-case polynomial-rate reduction of simple regret over time.

2.2 SEPARATION OF CONCERNS

If fast convergence to optimal choice is of interest, then Monte-Carlo planning should be as exploratory as possible [6]. However, what it means to be "as exploratory as possible" with MDPs is less straightforward than it is in MABs. In particular, recently it was observed that "forecasters" $s \in \mathcal{T}$ should be devoted to *two*, somewhat competing, *exploratory* objectives, namely identifying an optimal action $\pi^*(s)$, and estimating the value of that action, because this information is needed by the predecessor(s) of s in $\mathcal{T}$ [11].

Following this observation, Feldman and Domshlak [11] introduced MCTS2e, a refinement of MCTS scheme that implements the principle of "separation of concerns," whereby different parts of each sample are devoted to different exploration objectives. In MCTS2e (Figure 2a), rollouts are generated by a two-phase process in which the actions are selected according to an exploratory policy until an (iteration-specific) switching point, and from that point on, the actions are selected according to an estimation policy. The sub-routines END-OF-PROBE and UPDATE in MCTS2e are trivial, the UPDATE-NODE sub-routine extends this of MCTS with tree expansion, and, instead of ROLLOUT-POLICY of MCTS, specific instances of MCTS2e are achieved by instantiating three new sub-routines that determine the switching point for a rollout, and the action selection protocols for the two phases of the rollouts.

Feldman and Domshlak [11] show that a specific instance of MCTS2e, dubbed BRUE, achieves an *exponential-rate* reduction of simple regret over time, with the bounds on simple regret becoming meaningful after only exponential in H^2 number of samples. The specific MCTS2e sub-routines that define the BRUE algorithm are shown in Figure 2b. Similarly to UCT, each node/action pair (s, a) is associated with variables $n(s, a)$ and $\widehat{Q}(s, a)$, but with the latter being initialized to $-\infty$. BRUE instantiates MCTS2e by choosing actions uniformly at the exploration phase of the sample, choosing the best empirical actions at the estimation phase, and changing the switching point in a round-robin fashion over the entire horizon. Importantly, if the switching point of a rollout $\rho = \langle s_0, a_1, s_1, \ldots, a_H, s_H\rangle$ is σ, then only the state/action pair $(s_{\sigma-1}, a_\sigma)$ is updated by the information collected by ρ. That is, the information obtained by the estimation phase of ρ is used only for improving the estimate at state $s_{\sigma(n)-1}$, and is not pushed further up the sample. While that may appear wasteful and counterintuitive, this locality of update is required to satisfy the formal guarantees of BRUE on exponential-rate reduction of simple regret over time [11].

3 FAST OPTIMAL VS. FAST GOOD

A comparative evaluation on *Sailing* [18] and *PGame* [15] domains showed that BRUE is continually improving towards an optimal solution, rather quickly obtaining results better than UCT [11]. However, that evaluation also showed that UCT sometimes

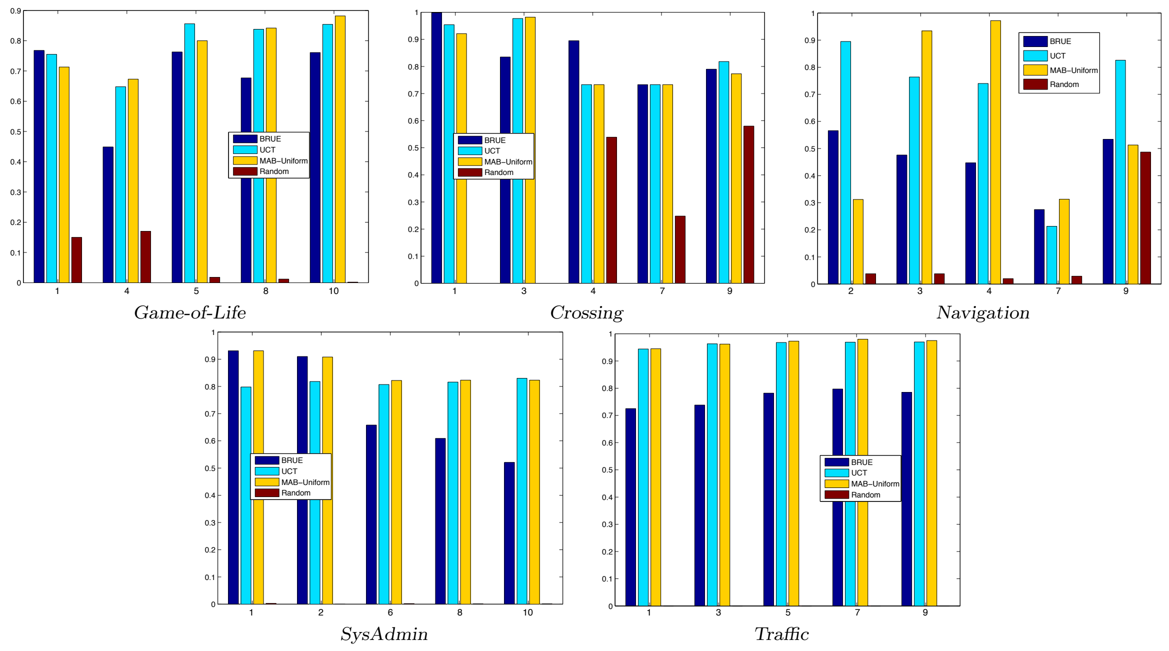


Figure 3: IPPC-2011 scores for the different MCTS algorithms under the (unknown to the algorithms) average deliberation time allowance of 5 seconds per step.

manages to identify reasonably good actions rather quickly, while BRUE is still "warming up". Focusing on tight deliberation deadlines, we conducted a wider empirical evaluation on MDP benchmarks from the last International Probabilistic Planning Competition (IPPC-2011). These benchmarks appear ideal for our purpose of evaluating algorithms under tight time constraints: Most IPPC-2011 domains induce very large branching factors, and thus allow only a very shallow sampling of the underlying search tree in reasonable time.

We used five IPPC-2011 domains, *Game-of-Life*, *SysAdmin*, *Traffic*, *Crossing*, and *Navigation*, with five randomly picked problem instances from each domain. Instances of *Game-of-Life*, *SysAdmin*, and *Traffic* were ran with planning horizon of 20 steps, whereas instances of (goal-driven) *Crossing*, and *Navigation* were ran with horizon of 40 steps. Each algorithm was allowed a (rather arbitrary chosen) deliberation time that started with 10 seconds for the first step, and decreased linearly to 1 second at the last step. In addition to UCT and BRUE, the comparison includes a trivial baseline of random action selection (Random), as well as MAB-Uniform, a simple algorithm that chooses actions everywhere uniformly at random, estimating $Q(s_0, a)$ for each $a \in A(s_0)$ by the value of taking a and then choosing actions uniformly at random for the remaining $H-1$ steps. Obviously, the estimates of MAB-Uniform are typically erroneous, and so a priori, its only positive property appears to be its computational efficiency.

The results are presented in Figure 3; the names of the problem instances on the x-axis are these from the IPPC-2011 repository, and y-axis captures the IPPC-2011 scoring scheme: The algorithms are scored relatively to each other on each problem instance, with the relative scores being then averaged over 300 runs on that specific problem instance. The relative score of a particular algorithm in a specific run is the difference between the total reward achieved by that algorithm and the worst total reward among all the algorithms, divided by the difference between such best and worst total rewards.

According to Figure 3, all the examined MCTS algorithms, including the seemingly naive MAB-Uniform, performed much better than Random, with *SysAdmin* and *Traffic* being the most prominent examples for that. In other words, even under severely limited deliberation time allowance, the value of deliberation in the benchmarks in use was substantial. Likewise, the experiment reconfirmed that UCT is typically more effective than BRUE under short deliberation times. This suggests that the "fast optimal" scheme of BRUE lacks some ingredients that make its MCTS competitors "fast good". Having said that, note that different problems favored different approaches, with MAB-Uniform being surprisingly superior on many of the examined benchmarks. Hence, the quest for a clear recipe for "fast good" remains open, especially if we do not want to neglect striving for optimality, and even more so, if we want that recipe to be robust on a wide palette of MDPs. This is precisely the quest we consider in what comes next.

4 TWO TYPES OF FORECASTERS

Consider the state/steps-to-go pairs (s, h) as a hierarchy of forecasters, all acting on behalf of the root forecaster (s_0, H) that aims at minimizing its own simple regret in a stochastic MAB induced by the applicable actions $A(s_0)$. In the setup of online planning, there is a conceptual difference between the exploration objective of the root forecaster and this of all other forecasters in the hierarchy. To see that, suppose there is an oracle that can provide each forecaster (s, h) *either* with the identity of the optimal action $\pi^*(s, h)$ but without revealing its value $Q_h(s, \pi^*(s, h))$, *or* with the value $Q_h(s, \pi^*(s, h))$ but without revealing the identity of $\pi^*(s, h)$. For the root forecaster (s_0, H), the first type of information is all he needs, while the second type of information buys him very little, if anything. In contrast, even if the oracle provides *all* the forecasters (s, h) *but* (s_0, H) with (only) the identities of the respective optimal actions $\pi^*(s, h)$, then the root forecaster (s_0, H) in some sense remains as clueless as it was before, and needs to explore the state space in order to obtain at least some ordinal information about the expected value of the alternative choices $A(s_0)$. However, if the oracle provides a non-root forecaster (s, h) (only) with the best Q-value among its alternative choices $A(s)$, then (s, h) can stop working since no further exploration of the sub-hierarchy rooted in (s, h) is needed.

In sum, what matters to the root forecaster is only what to execute, while all other forecaster care only about the value they can provide to their ancestors in the hierarchy, and *not about how this value can actually be acquired.* Of course, the reader may question this classification by arguing that these two objectives are just two sides of the same coin: estimating the value of optimal action assumes aiming at identifying an optimal action and vice versa. To some extent, that is true, but only to some extent. For instance, the very realization that this coin has two sides, and that these two sides are somewhat competing, is precisely what motivates the "separation of concerns" principle behind the MCTS2e scheme. Turns out that this classification of objectives suggests further insights into the dynamics of MCTS algorithms.

In all MCTS algorithms for online MDP planning, each iteration corresponds to examining a chain of forecasters within the overall hierarchy under (s_0, H), with the difference between the algorithm boiling down to two decisions:

(I) which chain of forecasters to examine, and

(II) how to estimate $Q_h(s, \pi^*(s, h))$ for each forecaster (s, h) in the hierarchy.

At first view, choosing the right strategy for (I) seems to be the key to rapid homing in on "good" decisions. The details of various MCTS algorithms suggest that their design was indeed primarily guided by choices for (I), with choices for (II) being implied by the former. Here, however, we suggest that decoupling these two decisions is important, and that the key to the quest of our interest actually lies in decision (II).

A closer look at different Monte-Carlo planning algorithms for MDPs reveals an interesting generalizing perspective on the way they all approach decision (II). Let $V_h^\pi(s)$ be the value of (s, h) under policy $\pi \in \mathbf{\Pi}$, $V_h^*(s) \equiv Q_h(s, \pi^*(s, h))$ be the value of (s, h) under the optimal policy, and let $\widehat{V}_h^\pi(s)$, $\widehat{V}_h^*(s)$ denote empirical estimates of these two quantities, respectively. In all MCTS algorithms, at each point of time, the entire hierarchy of forecasters can be seen as consisting of *two types* of forecasters.

$\mathrm{T_{OUT}}$ forecaster (s, h) (possibly schematically) estimate $V_h^*(s)$ by an estimate of $\mathbb{E}_{\pi \sim \mathcal{U}[\mathbf{\Pi}]} V_h^\pi(s)$, that is, of the expected total reward of a policy sampled from $\mathbf{\Pi}$ uniformly at random.

$\mathrm{T_{IN}}$ forecaster (s, h) distinguishes between its choices $A(s)$, and estimates $V_h^*(s)$ by an estimate of $\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) \left[R(s, a, s') + V_{h-1}^*(s')\right]$, where the estimate of $V_{h-1}^*(s')$ is based on the information provided by s' to s.

Consider the way in which the specific MCTS algorithms approach decision (II) in terms of this $\mathrm{T_{OUT}}/\mathrm{T_{IN}}$ partition of the forecasters. In both UCT and BRUE, *$\mathrm{T_{IN}}$-forecasters correspond to the nodes of $\mathcal{T}$, while all other state/steps-to-go pairs correspond to $\mathrm{T_{OUT}}$-forecasters.* Note that these $\mathrm{T_{OUT}}$-forecasters are very much not virtual. For instance, in UCT they are queried by the EVALUATE sub-routine, and in BRUE they are queried, possibly in interleaving with $\mathrm{T_{IN}}$-forecasters, by both EXPLORATION-POLICY and ESTIMATION-POLICY sub-routines.

At first view, $\mathrm{T_{OUT}}$-forecasters appear to be strangely lazy and potentially very misleading, while $\mathrm{T_{IN}}$-forecasters seem to be doing the right thing. However, it is not all that simple. First, while each $\mathrm{T_{OUT}}$-forecaster samples a single random variable, each $\mathrm{T_{IN}}$-forecaster (s, h) has to sample $|A(s)|$ random variables. Thus, $\mathrm{T_{OUT}}$-forecasters converge to quality estimates of quantities of their interest much faster than their $\mathrm{T_{IN}}$ counterparts. Second, while $\mathrm{T_{IN}}$-forecasters try to estimate the right thing, their success totally depends on the quality of estimates of $V_{h-1}^*(s')$ they receive from their successors. Hence, it is not clear that forecasters of type $\mathrm{T_{IN}}$ are always more effective.

We return to this issue in more detail later on. For now, note only that both UCT and BRUE can be seen as *continuously reconsidering the typing of the forecasters.* Specifically, in both UCT and BRUE, (at most) a single forecaster is "converted" from $\mathrm{T}_{\mathrm{OUT}}$ to T_{IN} at every iteration: in UCT it is the *shallowest* $\mathrm{T}_{\mathrm{OUT}}$-forecaster found along the rollout, and in BRUE, it is the $\mathrm{T}_{\mathrm{OUT}}$-forecaster that happens to lie at the rollout's switching point σ. This way, the set of T_{IN}-forecasters in UCT grows *incrementally* as a single community connected to the root forecaster (s_0, H). In contrast, T_{IN}-forecasters in BRUE evolve in H independent sets, each distributed over the respective depth level of the forecast hierarchy according to the transition distribution induced by the *uniform* action selection at the preceding levels.

This specific difference between UCT and BRUE is directly related to their relative efficiency under different orders of deliberation time allowance. Populating T_{IN}-forecasters at all levels of the hierarchy is generally necessary to guarantee fast convergence to optimal choice at the root. However, the marginal value of T_{IN}-forecasters at different levels vary with the deliberation time allowance: Information gathered by T_{IN}-forecasters at deep levels takes time to be propagated to the root, making their near-term influence on the choices at (s_0, H) smaller than this of the T_{IN}-forecasters closer to the root.

In that respect, a modification of BRUE that suggests itself almost immediately is as simple as it gets: Instead of converting the $\mathrm{T}_{\mathrm{OUT}}$-forecaster at the switching point σ, we can resort to converting the shallowest $\mathrm{T}_{\mathrm{OUT}}$-forecaster on the exploratory part of the rollout, that is, up to the level σ. By offering both exponential-rate reduction of the simple regret at the root, as well as incremental conversion of $\mathrm{T}_{\mathrm{OUT}}$-forecasters as a connected set around (s_0, H), the resulting algorithm, $\mathsf{BRUE}_{\mathcal{I}}$, substantially improves over BRUE in short-term effectiveness. (The specific empirical results for $\mathsf{BRUE}_{\mathcal{I}}$ are shown later in the paper.) However, this simple modification of BRUE is not our final destination, and next we show that this simple bridge between MCTS and MCTS2e opens a much wider window of opportunity.

5 SELECTIVE TREE EXPANSION

Similarly to UCT and BRUE, each iteration of $\mathsf{BRUE}_{\mathcal{I}}$ either finds no candidate for type conversion, or *unconditionally* converts a concrete single $\mathrm{T}_{\mathrm{OUT}}$-forecaster (s, h) to type T_{IN}. However, suppose we know that $\mathbb{E}_{\pi\sim\mathcal{U}[\mathbf{\Pi}]}V_h^{\pi}(s)$ *equals* $\max_{a\in A(s)}\sum_{s'} P(s' \mid s,a)\left[R(s,a,s') + V_{h-1}^{*}(s')\right]$. Since direct Monte-Carlo estimation of the quantity on the left-hand side is substantially easier than this of the right-hand side, converting (s, h) to type T_{IN} is clearly not a good idea. In fact, both (s, h) and all of its exclusive descendants in the hierarchy would better remain $\mathrm{T}_{\mathrm{OUT}}$-forecasters for the entire deliberation process, no matter how long it is. Of course, this equality rarely holds, and, more importantly, we have no prior knowledge about the size of the gap between the quality of the best policy under (s, h) and the expected quality of the randomly picked policy. However, this extreme example still hints on the promise of *selective* type conversion, and below we examine the prospects of this direction.

The variance of a Monte-Carlo estimator $\widehat{Q}_h(s, a)$ of the value of action a at state s stems from two sources. The first source of variance comes from following different policies (aka action selections) along different rollouts. The other source of variance comes from the stochastic nature of the action outcomes. That is, if r is the reward obtained by following policy π for h steps starting from state s, then

$$\mathrm{Var}\left[r\right] = \mathbb{E}\left[\mathrm{Var}\left[r \mid \pi\right]\right] + \mathrm{Var}\left[\mathbb{E}\left[r \mid \pi\right]\right]. \qquad (1)$$

At one extreme, we have all policies yielding the same expected reward, and thus all the variance comes from the action outcomes. In that case, distinguishing between the policies under (s, h) is not only useless, but also computationally harmful. Thus both (s, h) and its descendants should be left as type $\mathrm{T}_{\mathrm{OUT}}$, that is, not added to $\mathcal{T}$. At the other extreme, we have all actions being deterministic, but different policies yield very different reward. In that case, it may be valuable to convert (s, h) to T_{IN}, increasing the resolution at which the policies under (s, h) are examined. Unsurprisingly, in between these two extremes, the "value of conversion" is less straightforward. In the absence of any information about the stopping time, online planning algorithms should strive to do well under the assumption that termination point is near, while ensuring continuous improvement as more time is allowed. And as long as the precise mixture of these two desiderata remains vague, so remains the precise formulation of the value of conversion.

Having said that, as we do understand the high-level factors that affect the value of conversion, we can try estimating and combining these factors so to reflect the purported value of conversion. Here we propose and evaluate a simple and intuitive rule: *The candidate T_{OUT}-forecaster (s, h) should be converted iff the variance of the expected reward over different policies under (s, h) exceeds the average variance of the policies*, that is, iff

$$\mathrm{Var}\left[\mathbb{E}\left[\widehat{V}_h^{\pi}(s) \mid \pi\right]\right] \;>\; \mathbb{E}\left[\mathrm{Var}\left[\widehat{V}_h^{\pi}(s) \mid \pi\right]\right], \qquad (2)$$

which is equivalent to

$$\mathrm{Var}\left[\mathbb{E}\left[r \mid \pi\right]\right] > \mathbb{E}\left[\frac{\mathrm{Var}\left[r \mid \pi\right]}{n(s, \pi)}\right], \tag{3}$$

where r is the reward obtained by following policy π for h steps starting from state s, and $n(s, \pi)$ is the number of samples that induce the estimate $\widehat{V}_h^\pi(s)$.

The quantity on the left indicates the distance between the average quality of the policies and the quality of the optimal one, that is

$$\mathbb{E}_{\pi \sim \mathcal{U}[\mathbf{\Pi}]} V_h^\pi(s) - V_h^*(s). \tag{4}$$

The quantity on the right indicates the distance between the estimator and the mean of the policies value, where the division by $n(s, \pi)$ captures the effect of averaging on the variance. As the number of samples from each policy grows, estimators $\widehat{V}_h^\pi(s)$ approaches their means $V_h^\pi(s)$, and thus the gap captured by Eq. 4 becomes the dominant factor, in which case converting (s, h) to $\mathrm{T_{IN}}$ is estimated as valuable.

Based on this decision rule, we suggest a new instance of MCTS2e, $\mathsf{BRUE}_{\mathcal{IC}}$ (standing for BRUE with *incremental* and *selective* type conversion). The respective MCTS2e sub-routines of $\mathsf{BRUE}_{\mathcal{IC}}$ are depicted in Figure 4.

- Similarly to BRUE, the actions are selected uniformly at random at the exploration phase of the rollout, and the empirically best actions are selected at the estimation phase.
- Similarly to UCT and $\mathsf{BRUE}_{\mathcal{I}}$, the tree is expanded in an incremental fashion, maintaining the set of $\mathrm{T_{IN}}$ forecasters connected to the root.
- Unlike UCT and $\mathsf{BRUE}_{\mathcal{I}}$, after a forecaster (s, h) is added to the tree, it goes through an "evaluation period", and remains of type $\mathrm{T_{OUT}}$ until it passes that evaluation. Hence, the leaves of $\mathcal{T}$ consist of some $\mathrm{T_{IN}}$-forecasters, but also of all $\mathrm{T_{OUT}}$ candidates for $\mathrm{T_{IN}}$.

The evaluation of the "conditional candidates" (s, h) is captured by the CONVERT procedure depicted in Figure 5. This procedure estimates the two sources of variance at (s, h) (lines 1-4, explained below), and only when the variance of the policies' values exceeds the average variance of the policies value in the sense of Eq. 2 (the condition in line 5 fails), (s, h) is converted to a $\mathrm{T_{IN}}$-forecaster. At the actual conversion (lines 6-9), the information gathered at the evaluation period is used to initialize the standard node variables $\widehat{Q}(s, a)$ and $n(s, a)$.

```
END-OF-PROBE ($s$ : state, $d$ : depth)
  if $s \notin \mathcal{T}$ then add $s$ to $\mathcal{T}$
  if $n(s) > 0$ or CONVERT($s$) then return false
  if $d \le \sigma$ then
    $\sigma \leftarrow -1$          // dummy value, to prevent node update
    retract ← true
  return true
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
EVALUATE ($s$ : state, $d$ : depth)
  if $\left|\Pi_s^{\mathrm{A}}\right| < \phi$ then
    $\pi \leftarrow$ GENERATE-POLICY($s, H - d$)
    add $\pi$ to $\Pi_s^{\mathrm{A}}$     // and thus to $\Pi_s$
  else
    $\pi \sim \mathcal{U}\left[\Pi_s^{\mathrm{A}}\right]$
  for $t \leftarrow d \ldots H$ do
    $a \sim \pi(s, t)$
    $s' \sim P(S \mid s, a)$
    $r \leftarrow r + R(s, a, s')$
    $s \leftarrow s'$
  UPDATE-NODE($s, \pi, r$)
  if $\frac{\widehat{\mathrm{Var}}(s,\pi)}{n(s,\pi)} < \psi$ then remove $\pi$ from $\Pi_s^{\mathrm{A}}$
  return $r$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
UPDATE-NODE ($s$: state, $x$: action or policy, $r$ : reward)
  $n(s, x) \leftarrow n(s, x) + 1$
  $\delta \leftarrow r - \widehat{Q}(s, x)$
  $\widehat{Q}(s, x) \leftarrow \widehat{Q}(s, x) + \frac{\delta}{n(s,x)}$
  $\widehat{\mathrm{Var}}(s, x) \leftarrow \frac{\widehat{\mathrm{Var}}(s,x)\cdot(n(s,x)-2) + \delta\cdot\left(r - \widehat{Q}(s,x)\right)}{n(s,x)-1}$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
SWITCH-FUNCTION ($n$ : iteration, $\sigma \in [\![H]\!]$)
  if retract or $\sigma = H$ then $\sigma \leftarrow 0$
                   else $\sigma \leftarrow \sigma + 1$
  retract ← false
  return $\sigma$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
EXPLORATION-POLICY ($s$ : state)
  return $a \sim \mathcal{U}\left[A(s)\right]$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
ESTIMATION-POLICY ($s$ : state)
  return $a \sim \mathcal{U}\left[\{a \mid \arg\max_{a \in A(s)} \widehat{Q}(s, a)\}\right]$
```

Figure 4: MCTS2e sub-routines for $\mathsf{BRUE}_{\mathcal{IC}}$. For the CONVERT procedure called by END-OF-PROBE, see Figure 5.

Technically, candidate evaluation is performed as follows. For each $\mathrm{T_{IN}}$ candidate (s, h), the algorithm maintains a set of policies Π_s, as well as a subset of "active" policies $\Pi_s^{\mathrm{A}} \subseteq \Pi_s$, size of which is bounded by a fixed parameter ϕ. The set Π_s^{A} is used by the subroutine EVALUATE that selects a policy from Π_s^{A} for evaluation uniformly at random. In case ϕ allows expanding the active subset Π_s^{A}, EVALUATE uses a newly generated policy (e.g., by selecting actions uniformly at all states that can be reached by following the actions at preceding states). The new policy is then added to Π_s^{A} (and thus to Π_s).

After a policy π is selected by EVALUATE, it is exe-

```
CONVERT ($s$ : state)

1.  $m \leftarrow \sum_{\pi \in \Pi_s} n(s,\pi)$
2.  $\mathbf{EE} \leftarrow \sum_{\pi \in \Pi_s} \frac{n(s,\pi)}{m} \widehat{Q}(s,\pi)$
3.  $\mathbf{EV} \leftarrow \sum_{\pi \in \Pi_s} \frac{n(s,\pi)}{m} \widehat{\mathrm{Var}}(s,\pi)$
4.  $\mathbf{VE} \leftarrow \sum_{\pi \in \Pi_s} \frac{n(s,\pi)}{m} (\widehat{Q}(s,\pi) - \mathbf{EE})^2$

5.  if $\frac{\mathbf{EV}}{m} \geq \mathbf{VE}$ then
        return false
    else
6.      for $a \in A(s)$ do
7.          $\pi_a \leftarrow \arg\max_{\{\pi : \pi(s)=a\}} \widehat{Q}(s,\pi)$
8.          $\widehat{Q}(s,a) \leftarrow \widehat{Q}(s,\pi_a)$
9.          $n(s,a) \leftarrow n(s,\pi_a)$
        return true
```

Figure 5: Implementation of the decision rule of BRUE$_{\mathcal{IC}}$ for type conversion.

cuted once starting at (s,h). The resulting cumulative reward r is then used to update statistics about the particular policy π, including the empirical mean $\widehat{Q}(s,\pi)$ (which is just a more convenient for us naming for the estimator $\widehat{V}_h^\pi(s)$), the empirical variance $\widehat{\mathrm{Var}}(s,\pi)$, and the counter $n(s,\pi)$. The reward r is also used later on, in the recursive rollback of PROBE (Figure 2). When the variance of π decreases below a predefined threshold ψ, it is removed from the active subset Π_s^A to be replaced by a new policy. This mechanism ensures that all policies will eventually be sampled, which is essential to guarantee convergence of our estimates of the two sources of variance.

The sub-routine END-OF-PROBE bares similarity to this of UCT, but it is extended with enforcing T_{IN}-candidates to remain leaves as long as the respective CONVERT attempts come out unsuccessful. In CONVERT, the statistics about policies are used to estimate the mean of the policies' variance **EV**, the variance in the policies value **VE**, the overall mean value of the policies **EE**, and m, the total number of samples from all the policies in Π_s. If the evaluated T_{IN}-candidate (s,h) meats the conversion condition, CONVERT returns true, and the newborn T_{IN}-forecaster (s,h) is no longer forced to remain a leaf. Just before that, the standard node variables $\widehat{Q}(s,a)$ and $n(s,a)$ are initialized with the information gathered around the empirically best policy π_a that starts with the respective action a. Importantly, since the candidate evaluation criterion is not blocking, all candidates eventually convert to T_{IN} and act as in BRUE. This provides BRUE$_{\mathcal{IC}}$ with BRUE's quality convergence rate in long term.

At last, BRUE$_{\mathcal{IC}}$ borrows its incremental tree expansion mechanism from BRUE$_{\mathcal{I}}$: When a leaf is encountered before the switching point σ, no node is updated, and a global flag *retract* is set to true, signaling SWITCH-FUNCTION to set the switching point to 0 in the subsequent iteration. This way, the switching point is selected systematically as in BRUE, but now ranging between the root and a leaf rather than on the entire horizon.

Table 1: Aggregated IPPC-2011 scores from Figure 6; boldfacing indicates top performance in the respective domain (and overall, in the last row).

	MAB	UCT	BRUE	BRUE$_{\mathcal{I}}$	BRUE$_{\mathcal{IC}}$
Traffic	**0.97**	0.96	0.77	0.92	0.95
Crossing	0.83	0.84	0.85	0.83	**0.87**
Navigation	0.61	0.69	0.46	0.65	**0.75**
Game-of-Life	0.78	0.79	0.68	0.79	**0.82**
SysAdmin	0.86	0.81	0.73	0.86	**0.87**
total	0.81	0.82	0.70	0.81	**0.85**

This finalizes the description of BRUE$_{\mathcal{IC}}$. To examine its relative effectiveness, we conducted a comparative evaluation under the same experimental setup as described in Section 3. We evaluated five algorithms: MAB-Uniform, UCT, BRUE, BRUE$_{\mathcal{I}}$, and BRUE$_{\mathcal{IC}}$. The results are presented in Figure 3 and summarized in Table 1; all scores in Table 1 are within confidence bounds of ± 0.01. For the sake of readability, here we excluded Random from the presentation. Overall, BRUE$_{\mathcal{IC}}$ exhibited a robustly good performance across the domains, finishing top performer on most problem instances. BRUE$_{\mathcal{IC}}$ is also consistently better than BRUE$_{\mathcal{I}}$, indicating that selective type conversion plays an important role in its performance. We also notice that, in most cases, MAB-Uniform was not very far off from the leader, falling short only in few instances of *Navigation*. Since MAB-Uniform can be seen as BRUE$_{\mathcal{IC}}$ with an ultra-conservative, unsatisfiable condition for type conversion, this suggests that further introspection into the specific decision rule of BRUE$_{\mathcal{IC}}$, as well as into the impact of the parameters in its implementation, should be valuable.

Finally, we examine the simple regret reduction of BRUE$_{\mathcal{IC}}$ under varying time budgets, using the experimental settings for *Sailing* domain from [11]. In the bottom part of Figure 6, we plot the simple regret of the actions recommended by MAB-Uniform, UCT, BRUE, and BRUE$_{\mathcal{IC}}$ under different planning time windows, averaged over 300 instances of $10x10$ and $20x20$ grid sizes. The results reveal that not only does BRUE$_{\mathcal{IC}}$ significantly outperforms UCT uniformly over time, right from the beginning of deliberation when BRUE is still lagging behind, but also that its simple regret reduction rate is rather comparable to BRUE's in the longer term.

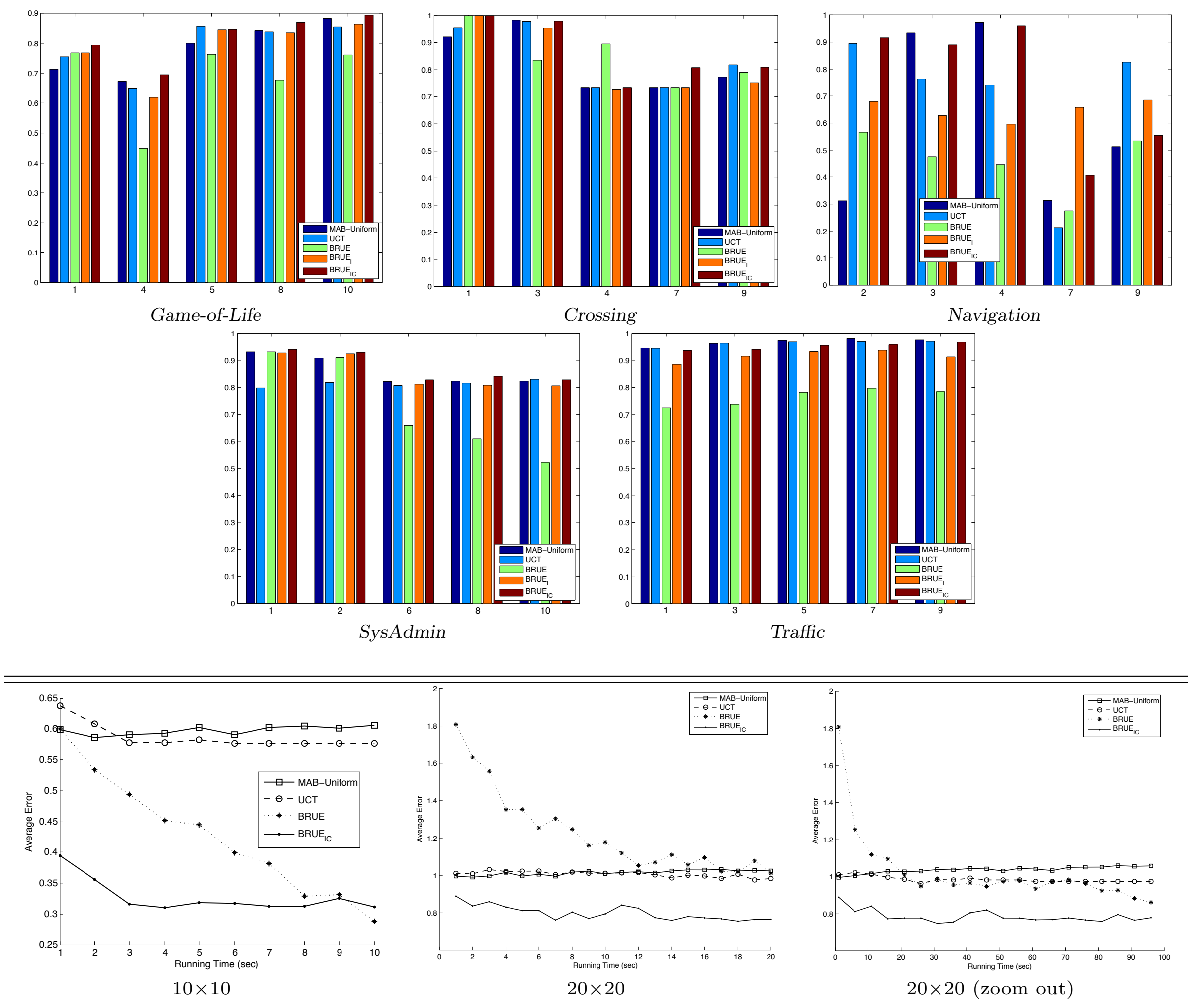


Figure 6: Performance of the MCTS algorithms with and without selective tree expansion. Top: Per-instance IPPC-2011 scores under the experimental setup as in Section 3. Bottom: Absolute performance of the algorithms in terms of average simple regret on the *Sailing* domain problems with 10×10 and 20×20 grid maps.

6 SUMMARY

Considering online planning for generative MDPs, we have investigated and combined the high-level principles that underly different computational schemes for this problem, and showed that their individual strengths can be put together at work. We then rationalized a principle of selective tree expansion that aims at automatically adapting Monte-Carlo exploration to the specifics of the MDP in hand, and suggested a concrete implementation of this principle within Monte-Carlo tree search methods. The resulting algorithm, $\mathsf{BRUE}_{\mathcal{IC}}$, favorably competes with other MCTS algorithms under short planning times, while preserving the attractive convergence properties of the (not so effective under short planning windows) algorithm BRUE [11], as well as the empirical strength of the latter under permissive planning windows.

In previous works, some forms of selective tree expansion has been shown extremely effective in *forward-search* planning for declarative MDPs [7, 4]. In that context, our work can be seen as the first selective tree expansion framework for Monte-Carlo planning, which is applicable in generative MDPs as well. Likewise, our work joins some other recent techniques for enhancing Monte-Carlo planning with adaptivity to the given problem, such as hindsight optimization for declarative MDPs [17, 26], and metalevel decision procedures for sample routing for both declarative and generative MDPs [13]. At the high level, these techniques appear complementary to selective tree expansion, and thus studying their interplay and cumulative value is a promising venue for future research.

Acknowledgements This work was partially supported by the EOARD grant FA8655-12-1-2096, and the ISF grant 1045/12.

References

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

[2] R. Balla and A. Fern. UCT for tactical assault planning in real-time strategy games. In *IJCAI*, pages 40–45, 2009.

[3] R. Bjarnason, A. Fern, and P. Tadepalli. Lower bounding Klondike Solitaire with Monte-Carlo planning. In *ICAPS*, 2009.

[4] B. Bonet and H. Geffner. Action selection for MDPs: Anytime AO* vs. UCT. In *AAAI*, 2012.

[5] S. Bubeck and R. Munos. Open loop optimistic planning. In *COLT*, pages 477–489, 2010.

[6] S. Bubeck, R. Munos, and G. Stoltz. Pure exploration in finitely-armed and continuous-armed bandits. *Theor. Comput. Sci.*, 412(19):1832–1852, 2011.

[7] L. Busoniu and R. Munos. Optimistic planning for Markov decision processes. In *AISTATS*, number 22 in JMLR (Proceedings Track), pages 182–189, 2012.

[8] T. Cazenave. Nested Monte-Carlo search. In *IJCAI*, pages 456–461, 2009.

[9] P-A. Coquelin and R. Munos. Bandit algorithms for tree search. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 67–74, Vancouver, BC, Canada, 2007.

[10] P. Eyerich, T. Keller, and M. Helmert. High-quality policies for the Canadian Traveler's problem. In *AAAI*, 2010.

[11] Z. Feldman and C. Domshlak. Simple regret optimization in online planning for markov decision processes. *CoRR*, `arXiv:1206.3382v2 [cs.AI]`, 2012.

[12] S. Gelly and D. Silver. Monte-Carlo tree search and rapid action value estimation in computer Go. *AIJ*, 175(11):1856–1875, 2011.

[13] N. Hay, S. E. Shimony, D. Tolpin, and S. Russell. Selecting computations: Theory and applications. In *UAI*, 2012.

[14] T. Keller and P. Eyerich. Probabilistic planning based on UCT. In *ICAPS*, 2012.

[15] L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo planning. In *ECML*, pages 282–293, 2006.

[16] A. Kolobov, Mausam, and D. Weld. LRTDP vs. UCT for online probabilistic planning. In *AAAI*, 2012.

[17] A. Y. Ng and M. Jordan. PEGASUS: A policy search method for large MDPs and POMDPs. In *UAI*, 2000.

[18] L. Péret and F. Garcia. On-line search for solving Markov decision processes via heuristic sampling. In *ECAI*, pages 530–534, 2004.

[19] M. Puterman. *Markov Decision Processes.* Wiley, 1994.

[20] H. Robbins. Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc.*, 58(5):527535, 1952.

[21] C. D. Rosin. Nested rollout policy adaptation for Monte Carlo tree search. In *IJCAI*, pages 649–654, 2011.

[22] N. Sturtevant. An analysis of UCT in multi-player games. In *CCG*, page 3749, 2008.

[23] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, 1998.

[24] D. Tolpin and S. E. Shimony. Doing better than UCT: Rational Monte Carlo sampling in trees. *CoRR*, arXiv:1108.3711v1 [cs.AI], 2011.

[25] D. Tolpin and S. E. Shimony. MCTS based on simple regret. In *AAAI*, 2012.

[26] S. W. Yoon, A. Fern, R. Givan, and S. Kambhampati. Probabilistic planning via determinization in hindsight. In *AAAI*, pages 1010–1016, 2008.

Bethe-ADMM for Tree Decomposition based Parallel MAP Inference

Qiang Fu Huahua Wang Arindam Banerjee
Department of Computer Science & Engineering
University of Minnesota, Twin Cities
Minneapolis, MN 55455
{qifu, huwang, banerjee}@cs.umn.edu

Abstract

We consider the problem of maximum a posteriori (MAP) inference in discrete graphical models. We present a parallel MAP inference algorithm called Bethe-ADMM based on two ideas: tree-decomposition of the graph and the alternating direction method of multipliers (ADMM). However, unlike the standard ADMM, we use an inexact ADMM augmented with a Bethe-divergence based proximal function, which makes each subproblem in ADMM easy to solve in parallel using the sum-product algorithm. We rigorously prove global convergence of Bethe-ADMM. The proposed algorithm is extensively evaluated on both synthetic and real datasets to illustrate its effectiveness. Further, the parallel Bethe-ADMM is shown to scale almost linearly with increasing number of cores.

1 Introduction

Given a discrete graphical model with known structure and parameters, the problem of finding the most likely configuration of the states is known as the *maximum a posteriori* (MAP) inference problem [23]. Existing approaches to solving MAP inference problems on graphs with cycles often consider a graph-based linear programming (LP) relaxation of the integer program [4, 18, 22] .

To solve the graph-based LP relaxation problem, two main classes of algorithms have been proposed. The first class of algorithms are dual LP algorithms [7, 9, 11, 19, 20, 21], which uses the dual decomposition and solves the dual problem. The two main approaches to solving the dual problems are block coordinate descent [7] and sub-gradient algorithms [11]. The coordinate descent algorithms are empirically faster, however, they may not reach the dual optimum since the dual problem is not strictly convex. Recent advances in coordinate descent algorithms perform tree-block updates [20, 21]. The sub-gradient methods, which are guaranteed to converge to the global optimum, can be slow in practice. For a detailed discussion on dual MAP algorithms, we refer the readers to [19]. The second class of algorithms are primal LP algorithms like the proximal algorithm [18]. The advantage of such algorithms is that it can choose different Bregman divergences as proximal functions which can take the graph structure into account. However, the proximal algorithms do not have a closed form update at each iteration in general and thus lead to double-loop algorithms.

As solving MAP inference in large scale graphical models is becoming increasingly important, in recent work, parallel MAP inference algorithms [14, 15] based on the alternating direction method of multipliers (ADMM) [1] have been proposed. As a primal-dual algorithm, ADMM combines the advantage of dual decomposition and the method of multipliers, which is guaranteed to converge globally and at a rate of $O(1/T)$ even for non-smooth problems [24]. ADMM has also been successfully used to solve large scale problem in a distributed manner [1].

Design of efficient parallel algorithms based on ADMM by problem decomposition has to consider a key tradeoff between the number of subproblems and the size of each subproblem. Having several simple subproblems makes solving each problem easy, but one has to maintain numerous dual variables to achieve consensus. On the other hand, having a few subproblems makes the number of constraints small, but each subproblem needs an elaborate often iterative algorithm, yielding a double-loop. Existing ADMM based algorithms for MAP inference [14, 15] decompose the problem into several simple subproblems, often based on single edges or local factors, so that the subproblems are easy to solve. However, to enforce consensus among the shared variables, such methods have to use dual variables proportional to the number of edges or local factors, which can make convergence slow on large graphs.

To overcome the limitations of existing ADMM methods for MAP inference, we propose a novel parallel algorithm based on tree decomposition. The individual trees need not be spanning and thus includes both edge decomposition and spanning tree decomposition as special cases. Compared to edge decomposition, tree decomposition has the flexibility of increasing the size of subproblems and reducing the number of subproblems by considering the graph structure. Compared to the tree block coordinate descent [20], which works with one tree at a time, our algorithm updates all trees in parallel. Note that the tree block coordinate descent algorithm in [21] updates disjoint trees within a forest in parallel, whereas our updates consider overlapping trees in parallel.

However, tree decomposition raises a new problem: the subproblems cannot be solved efficiently in the ADMM framework and requires an iterative algorithm, yielding a double-loop algorithm [14, 18]. To efficiently solve the subproblem on a tree, we propose a novel inexact ADMM algorithm called Bethe-ADMM, which uses a Bregman divergence induced by Bethe entropy on a tree, instead of the standard quadratic divergence, as the proximal function. The resulting subproblems on each tree can be solved exactly in linear time using the sum-product algorithm [12]. However, the proof of convergence for the standard ADMM does not apply to Bethe-ADMM. We prove global convergence of Bethe-ADMM and establish a $O(1/T)$ convergence rate, which is the same as the standard ADMM [8, 24]. Overall, Bethe-ADMM overcomes the limitations of existing ADMM based MAP inference algorithms [14, 15] and provides the flexibility required in designing efficient parallel algorithm through: (i) Tree decomposition, which can take the graph structure into account and greatly reduce the number of variables participating in the consensus and (ii) the Bethe-ADMM algorithm, which yields efficient updates for each subproblem.

We compare the performance of Bethe-ADMM with existing methods on both synthetic and real datasets and illustrate four aspects. First, Bethe-ADMM is faster than existing primal LP methods in terms of convergence. Second, Bethe-ADMM is competitive with existing dual methods in terms of quality of solutions obtained. Third, in certain graphs, tree decomposition leads to faster convergence than edge decomposition for Bethe-ADMM. Forth, parallel Bethe-ADMM, based on Open MPI, gets substantial speedups over sequential Bethe-ADMM. In particular, we show almost linear speed-ups with increasing number of cores on a graph with several million nodes.

The rest of the paper is organized as follows: We review the MAP inference problem in Section 2. In Section 3, we introduce the Bethe-ADMM algorithm and prove its global convergence. We discuss empirical evaluation in Section 4, and conclude in Section 5.

2 Background and Related Work

We first introduce some basic background on Markov Random Fields (MRFs). Then we briefly review existing ADMM based MAP inference algorithms in the literature. We mainly focus on pairwise MRFs and the discussions can be easily carried over to MRFs with general factors.

2.1 Problem Definition

A pairwise MRF is defined on an undirected graph $G = (V, E)$, where V is the vertex set and E is the edge set. Each node $u \in V$ has a random variable X_u associated with it, which can take value x_u in some discrete space $\mathcal{X} = \{1, \ldots, k\}$. Concatenating all the random variables $X_u, \forall u \in V$, we obtain an n dimensional random vector $\boldsymbol{X} = \{X_u | u \in V\} \in \mathcal{X}^n$. We assume that the distribution P of $\boldsymbol{X}$ is a Markov Random Field [23], meaning that it factors according to the structure of the undirected graph G as follows: With $f_u : \mathcal{X} \mapsto \mathbb{R}, \forall u \in V$ and $f_{uv} : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, $\forall (u,v) \in E$ denoting nodewise and edgewise potential functions respectively, the distribution takes the form $P(\boldsymbol{x}) \propto \exp\left\{\sum_{u\in V} f_u(x_u) + \sum_{(u,v)\in E} f_{uv}(x_u, x_v)\right\}$.

An important problem in the context of MRF is that of *maximum a posteriori* (MAP) inference, which is the following integer programming (IP) problem:

$$\boldsymbol{x}^* \in \operatorname*{argmax}_{\boldsymbol{x}\in\mathcal{X}^n} \left\{ \sum_{u\in V} f_u(x_u) + \sum_{(u,v)\in E} f_{uv}(x_u, x_v) \right\} . \tag{1}$$

The complexity of (1) depends critically on the structure of the underlying graph. When G is a tree structured graph, the MAP inference problem can be solved efficiently via the max-product algorithm [12]. However, for an arbitrary graph G, the MAP inference algorithm is usually computationally intractable. The intractability motivates the development of algorithms to solve the MAP inference problem approximately. In this paper, we focus on the linear programming (LP) relaxation method [4, 22]. The LP relaxation of MAP inference problem is defined on a set of pseudomarginals μ_u and μ_{uv}, which are non-negative, normalized and locally consistent [4, 22]:

$$\begin{aligned} \mu_u(x_u) &\geq 0 , \quad \forall u \in V , \\ \sum_{x_u\in\mathcal{X}_u} \mu_u(x_u) &= 1, \quad \forall u \in V , \\ \mu_{uv}(x_u, x_v) &\geq 0, \quad \forall (u,v) \in E , \\ \sum_{x_u\in\mathcal{X}_u} \mu_{uv}(x_u, x_v) &= \mu_v(x_v), \quad \forall (u,v) \in E . \end{aligned} \tag{2}$$

We denote the polytope defined by (2) as $L(G)$. The LP relaxation of MAP inference problem (1) becomes solving the following LP:

$$\max_{\boldsymbol{\mu}\in L(G)} \langle \boldsymbol{\mu}, \boldsymbol{f} \rangle . \tag{3}$$

If the solution $\boldsymbol{\mu}$ to (3) is an integer solution, it is guaranteed to be the optimal solution of (1). Otherwise, one can apply rounding schemes [17, 18] to round the fractional solution to an integer solution.

2.2 ADMM based MAP Inference Algorithms

In recent years, ADMM [14, 15] has been used to solve large scale MAP inference problems. To solve (3) using ADMM, we need to split nodes or/and edges and introduce equality constraints to enforce consensus among the shared variables. The algorithm in [14] adopts edge decomposition and introduces equality constraints for shared nodes. Let d_i be the degree of node i. The number of equality constraints in [14] is $O(\sum_{i=1}^{|V|} d_i k)$, which is approximately equal to $O(|E|k)$. For binary pairwise MRFs, the subproblems for the ADMM in [14] have closed-form solutions. For multi-valued MRFs, however, one has to first binarize the MRFs which introduces additional $|V|k$ variables for nodes and $2|E|k^2$ variables for edges. The binarization process increases the number of factors to $O(|V| + 2|E|k)$ and the complexity of solving each subproblem increases to $O(|E|k^2 \log k)$. We note that in a recent work [13], the active set method is employed to solve the quadratic problem for arbitrary factors. A generalized variant of [14] which does not require binarization is presented in [15]. We refer to this algorithm as Primal ADMM and use it as a baseline in Section 4. Although each subproblem in primal ADMM can be efficiently solved, the number of equality constraints and dual variables is $O(2|E|k + |E|k^2)$. In [15], ADMM is also used to solve the dual of (1). We refer to this algorithm as the Dual ADMM algorithm and use it as a baseline in Section 4. The dual ADMM works for multi-valued MRFs and has a linear time algorithm for each subproblem, but the number of equality constraint is $O(2|E|k + |E|k^2)$.

3 Algorithm and Analysis

We first show how to solve (3) using ADMM based on tree decomposition. The resulting algorithm can be a double-loop algorithm since some updates do not have closed form solutions. We then introduce the Bethe-ADMM algorithm where every subproblem can be solved exactly and efficiently, and analyze its convergence properties.

3.1 ADMM for MAP Inference

We first show how to decompose (3) into a series of subproblems. We can decompose the graph G into overlapping subgraphs and rewrite the optimization problem with consensus constraints to enforce the pseudo-marginals on subgraphs (local variables) to agree with $\boldsymbol{\mu}$ (global variable). Throughout the paper, we focus on tree-structured decompositions. To be more specific, let $\mathbb{T} = \{(V_1, E_1), \ldots, (V_{|\mathbb{T}|}, E_{|\mathbb{T}|})\}$ be a collection of subgraphs of G which satisfies two criteria: (i) Each subgraph $\tau = (V_\tau, E_\tau)$ is a tree-structured graph and (ii) Each node $u \in V$ and each edge $(u, v) \in E$ is included in at least one subgraph $\tau \in \mathbb{T}$. We also introduce local variable $\boldsymbol{m}_\tau \in L(\tau)$ which is the pseudomarginal [4, 22] defined on each subgraph τ. We use $\boldsymbol{\theta}_\tau$ to denote the potentials on subgraph τ. We denote $\boldsymbol{\mu}_\tau$ as the components of global variable $\boldsymbol{\mu}$ that belong to subgraph τ. Note that since $\boldsymbol{\mu} \in L(G)$ and τ is a tree-structured subgraph of G, $\boldsymbol{\mu}_\tau$ always lies in $L(\tau)$. In the newly formulated optimization problem, we will impose consensus constraints for shared nodes and edges. For the ease of exposition, we simply use the equality constraint $\boldsymbol{\mu}_\tau = \boldsymbol{m}_\tau$ to enforce the consensus.

The new optimization problem we formulate based on graph decomposition is then as follows:

$$\min_{\boldsymbol{m}_\tau, \boldsymbol{\mu}} \quad \sum_{\tau=1}^{|\mathbb{T}|} \rho_\tau \langle \boldsymbol{m}_\tau, \boldsymbol{\theta}_\tau \rangle \tag{4}$$

$$\text{subject to} \quad \boldsymbol{m}_\tau - \boldsymbol{\mu}_\tau = 0, \quad \tau = 1, \ldots, |\mathbb{T}| \tag{5}$$

$$\boldsymbol{m}_\tau \in L(\tau), \quad \tau = 1, \ldots, |\mathbb{T}| \tag{6}$$

where ρ_τ is a positive constant associated with each subgraph. We use the consensus constraints (5) to make sure that the pseudomarginals agree with each other in the shared components across all the tree-structured subgraphs. Besides the consensus constraints, we also impose feasibility constraints (6), which guarantee that, for each subgraph, the local variable $\boldsymbol{m}_\tau$ lies in $L(\tau)$. When the constraints (5) and (6) are satisfied, the global variable $\boldsymbol{\mu}$ is guaranteed to lie in $L(G)$.

To make sure that problem (3) and (4)-(6) are equivalent, we also need to guarantee that

$$\min_{\boldsymbol{m}_\tau} \sum_{\tau=1}^{|\mathbb{T}|} \rho_\tau \langle \boldsymbol{m}_\tau, \boldsymbol{\theta}_\tau \rangle = \max_{\boldsymbol{\mu}} \langle \boldsymbol{\mu}, \boldsymbol{f} \rangle \ , \tag{7}$$

assuming the constraints (5) and (6) are satisfied. It is easy to verify that, as long as (7) is satisfied, the specific choice of ρ_τ and $\boldsymbol{\theta}_\tau$ do not change the problem. Let $\mathbf{1}[.]$ be a binary indicator function and $\boldsymbol{l} = -\boldsymbol{f}$. For any positive $\rho_\tau, \forall \tau \in \mathbb{T}$, e.g., $\rho_\tau = 1$, a simple approach to obtaining the potential $\boldsymbol{\theta}_\tau$ can be:

$$\theta_{\tau,u}(x_u) = \frac{l_u(x_u)}{\sum_{\tau'} \rho_{\tau'} \mathbf{1}[u \in V_{\tau'}]}, u \in V_\tau \ ,$$

$$\theta_{\tau,uv}(x_u, x_v) = \frac{l_{uv}(x_u, x_v)}{\sum_{\tau'} \rho_{\tau'} \mathbf{1}[(u,v) \in E_{\tau'}]}, (u,v) \in E(\tau) \ .$$

Let $\boldsymbol{\lambda}_\tau$ be the dual variable and $\beta > 0$ be the penalty parameter. The following updates constitute a single iteration of the ADMM [1]:

$$\boldsymbol{m}_\tau^{t+1} = \operatorname*{argmin}_{\boldsymbol{m}_\tau \in L(\tau)} \langle \boldsymbol{m}_\tau, \rho_\tau \boldsymbol{\theta}_\tau + \boldsymbol{\lambda}_\tau^t \rangle + \frac{\beta}{2} ||\boldsymbol{m}_\tau - \boldsymbol{\mu}_\tau^t||_2^2 \ , \tag{8}$$

$$\boldsymbol{\mu}^{t+1} = \operatorname*{argmin}_{\boldsymbol{\mu}} \sum_{\tau=1}^{|\mathbb{T}|} \left(-\langle \boldsymbol{\mu}_\tau, \boldsymbol{\lambda}_\tau^t \rangle + \frac{\beta}{2} ||\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau||_2^2 \right) , \tag{9}$$

$$\boldsymbol{\lambda}_\tau^{t+1} = \boldsymbol{\lambda}_\tau^t + \beta(\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^{t+1}) \ . \tag{10}$$

In the tree based ADMM (8)-(10), the equality constraints are only required for shared nodes and edges. Assume there are m shared nodes and the shared node v_i has C_i^v copies and there are n shared edges and the shared edge e_j has C_j^e copies. The total number of equality constraints is $O(\sum_{i=1}^m C_i^v k + \sum_{j=1}^n C_j^e k^2)$. A special case of tree decomposition is edge decomposition, where only nodes are shared. In edge decomposition, $n = 0$ and the number of equality constraints is $O(\sum_{i=1}^m C_i^v k)$, which is approximately equal to $O(|E|k)$ and similar to [14]. In general, the number of shared nodes and edges in tree decomposition is much smaller than that in edge decomposition. The smaller number of equality constraints usually lead to faster convergence in achieving consensus. Now, the problem turns to whether the updates (8) and (9) can be solved efficiently, which we analyze below:

Updating $\boldsymbol{\mu}$: Since we have an unconstrained optimization problem (9) and the objective function decomposes component-wisely, taking the derivatives and setting them to zero yield the solution. In particular, let S_u be the set of subgraphs which contain node u, for the node components, we have:

$$\mu_u^{t+1}(x_u) = \frac{1}{|S_u|\beta} \sum_{\tau \in S_u} \left(\beta m_{\tau,u}^{t+1}(x_u) + \lambda_{\tau,u}^t(x_u) \right) . \quad (11)$$

(11) can be further simplified by observing that $\sum_{\tau \in S_u} \lambda_{\tau,u}^t(x_u) = 0$ [1]:

$$\mu_u^{t+1}(x_u) = \frac{1}{|S_u|} \sum_{\tau=1}^{T} m_{\tau,u}^{t+1}(x_u) . \quad (12)$$

Let S_{uv} be the subgraphs which contain edge (u, v). The update for the edge components can be similarly derived as:

$$\mu_{u,v}^{t+1}(x_u, x_v) = \frac{1}{|S_{uv}|} \sum_{\tau \in S_{uv}} m_{\tau,uv}^{t+1}(x_u, x_v) . \quad (13)$$

Updating $\boldsymbol{m}_\tau$: For (8), we need to solve a quadratic optimization problem for each tree-structured subgraph. Unfortunately, we do not have a close-form solution for (8) in general. One possible approach, similar to the proximal algorithm, is to first obtain the solution $\tilde{\boldsymbol{m}}_\tau$ to the unconstrained problem of (8) and then project $\tilde{\boldsymbol{m}}_\tau$ to $L(\tau)$:

$$\boldsymbol{m}_\tau = \underset{\boldsymbol{m} \in L(\tau)}{\operatorname{argmin}} ||\boldsymbol{m} - \tilde{\boldsymbol{m}}_\tau||_2^2 . \quad (14)$$

If we adopt the cyclic Bregman projection algorithm [2] to solve (14), the algorithm becomes a double-loop algorithm, i.e., the cyclic projection algorithm projects the solution to each individual constraint of $L(\tau)$ until convergence and the projection algorithm itself is iterative. We refer to this algorithm as the Exact ADMM and use it as a baseline in Section 4.

3.2 Bethe-ADMM

Instead of solving (8) exactly, a common way in inexact ADMMs [10, 25] is to linearize the objective function in (8), i.e., the first order Taylor expansion at $\boldsymbol{m}_\tau^t$, and add a new quadratic penalty term such that

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\operatorname{argmin}} \langle \mathbf{y}_\tau^t, \boldsymbol{m}_\tau - \boldsymbol{m}_\tau^t \rangle + \frac{\alpha}{2} \|\boldsymbol{m}_\tau - \boldsymbol{m}_\tau^t\|_2^2 , \quad (15)$$

where α is a positive constant and

$$\mathbf{y}_\tau^t = \rho_\tau \boldsymbol{\theta}_\tau + \boldsymbol{\lambda}_\tau^t + \beta(\boldsymbol{m}_\tau^t - \boldsymbol{\mu}_\tau^t) . \quad (16)$$

However, as discussed in the previous section, the quadratic problem (15) is generally difficult for a tree-structured graph and thus the conventional inexact ADMM does not lead to an efficient update for $\boldsymbol{m}_\tau$. By taking the tree structure into account, we propose an inexact minimization of (8) augmented with a Bregman divergence induced by the Bethe entropy. We show that the resulting proximal problem can be solved exactly and efficiently using the sum-product algorithm [12]. We prove that the global convergence of the Bethe-ADMM algorithm in Section 3.3.

The basic idea in the new algorithm is that we replace the quadratic term in (15) with a Bregman-divergence term $d_\phi(\boldsymbol{m}_\tau || \boldsymbol{m}_\tau^t)$ such that

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\operatorname{argmin}} \langle \mathbf{y}_\tau^t, \boldsymbol{m}_\tau - \boldsymbol{m}_\tau^t \rangle + \alpha d_\phi(\boldsymbol{m}_\tau || \boldsymbol{m}_\tau^t) , \quad (17)$$

is efficient to solve for any tree τ. Expanding the Bregman divergence and removing the constants, we can rewrite (17) as

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\operatorname{argmin}} \langle \mathbf{y}_\tau^t / \alpha - \nabla\phi(\boldsymbol{m}_\tau^t), \boldsymbol{m}_\tau \rangle + \phi(\boldsymbol{m}_\tau). \quad (18)$$

For a tree-structured problem, what convex function $\phi(\boldsymbol{m}_\tau)$ should we choose? Recall that $\boldsymbol{m}_\tau$ defines the marginal distributions of a tree-structured distribution $p_{\boldsymbol{m}_\tau}$ over the nodes and edges:

$$\boldsymbol{m}_{\tau,u}(x_u) = \sum_{\neg x_u} p_{\boldsymbol{m}_\tau}(x_1, \ldots, x_u, \ldots, x_n), \ \forall u \in V_\tau ,$$

$$\boldsymbol{m}_{\tau,uv}(x_u, x_v) = \sum_{\neg x_u, \neg x_v} p_{\boldsymbol{m}_\tau}(x_1, \ldots, x_u, x_v, \ldots, x_n), \ \forall (uv) \in E_\tau .$$

It is well known that the sum-product algorithm [12] efficiently computes the marginal distributions for a tree structured graph. It can also be shown that the sum-product algorithm solves the following optimization problem [23] for tree τ for some constant $\boldsymbol{\eta}_\tau$:

$$\max_{\boldsymbol{m}_\tau \in L(\tau)} \langle \boldsymbol{m}_\tau, \boldsymbol{\eta}_\tau \rangle + H_{Bethe}(\boldsymbol{m}_\tau) , \quad (19)$$

where $H_{Bethe}(\boldsymbol{m}_\tau)$ is the Bethe entropy of $\boldsymbol{m}_\tau$ defined as:

$$H_{Bethe}(\boldsymbol{m}_\tau) = \sum_{u \in V_\tau} H_u(\boldsymbol{m}_{\tau,u}) - \sum_{(u,v) \in E_\tau} I_{uv}(\boldsymbol{m}_{\tau,uv}) , \quad (20)$$

where $H_u(\boldsymbol{m}_{\tau,u})$ is the entropy function on each node $u \in V_\tau$ and $I_{uv}(\boldsymbol{m}_{\tau,uv})$ is the mutual information on each edge $(u,v) \in E_\tau$.

Combing (18) and (19), we set $\boldsymbol{\eta}_\tau = \nabla\phi(\boldsymbol{m}_\tau^t) - \mathbf{y}_\tau^t/\alpha$ and choose ϕ to be the negative Bethe entropy of $\boldsymbol{m}_\tau$ so that (18) can be solved efficiently in linear time via the sum-product algorithm.

For the sake of completeness, we summarize the Bethe-ADMM algorithm as follows :

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\operatorname{argmin}} \langle \mathbf{y}_\tau^t/\alpha - \nabla\phi(\boldsymbol{m}_\tau^t), \boldsymbol{m}_\tau \rangle + \phi(\boldsymbol{m}_\tau) \ , \quad (21)$$

$$\boldsymbol{\mu}^{t+1} = \underset{\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{\tau=1}^{T} \left(-\langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau \rangle + \frac{\beta}{2} ||\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau||_2^2 \right), (22)$$

$$\boldsymbol{\lambda}_\tau^{t+1} = \boldsymbol{\lambda}_\tau^t + \beta(\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^{t+1}) \ , \quad (23)$$

where $\mathbf{y}_\tau^t$ is defined in (16) and $-\phi$ is defined in (20).

3.3 Convergence

We prove the global convergence of the Bethe-ADMM algorithm. We first bound the Bregman divergence d_ϕ:

Lemma 1 *Let $\boldsymbol{\mu}_\tau$ and $\boldsymbol{\nu}_\tau$ be two concatenated vectors of the pseudomarginals on a tree τ with n_τ nodes. Let $d_\phi(\boldsymbol{\mu}_\tau||\boldsymbol{\nu}_\tau)$ be the Bregman divergence induced by the negative Bethe entropy ϕ. Assuming $\alpha \geq \max_\tau\{\beta(2n_\tau - 1)^2\}$, we have*

$$\alpha d_\phi(\boldsymbol{\mu}_\tau||\boldsymbol{\nu}_\tau) \geq \frac{\beta}{2}\|\boldsymbol{\mu}_\tau - \boldsymbol{\nu}_\tau\|_2^2 \ . \quad (24)$$

Proof: Let $P_\tau(\boldsymbol{x})$ be a tree-structured distribution on a tree $\tau = (V_\tau, E_\tau)$, where $|V_\tau| = n_\tau$ and $|E_\tau| = n_\tau - 1$. The pseudomarginal $\boldsymbol{\mu}_\tau$ has a total of $2n_\tau - 1$ components, each being a marginal distribution. In particular, there are n_τ marginal distributions corresponding to each node $u \in V_\tau$, given by

$$\mu_{\tau,u}(x_u) = \sum_{\neg x_u} P_\tau(x_1, \ldots, x_u, \ldots, x_n) \ . \quad (25)$$

Thus, $\boldsymbol{\mu}_u$ is the marginal probability for node u.

Further, there are $n_\tau - 1$ marginal components corresponding to each edge $(u,v) \in E_\tau$, given by

$$\mu_{\tau,uv}(x_u, x_v) = \sum_{\neg(x_u,x_v)} P(x_1, \ldots, x_u, \ldots, x_v, \ldots, x_n) \ . \quad (26)$$

Thus, $\boldsymbol{\mu}_{uv}$ is the marginal probability for nodes (u,v).

Let $\boldsymbol{\mu}_\tau, \boldsymbol{\nu}_\tau$ be two pseudomarginals defined on tree τ and $P_{\boldsymbol{\mu}_\tau}, P_{\boldsymbol{\nu}_\tau}$ be the corresponding tree-structured distributions. Making use of (25), we have

$$\|P_{\boldsymbol{\mu}_\tau} - P_{\boldsymbol{\nu}_\tau}\|_1 \geq \|\boldsymbol{\mu}_{\tau,u} - \boldsymbol{\nu}_{\tau,u}\|_1, \quad \forall u \in V_\tau \ . \quad (27)$$

Similarly, for each edge, we have the following inequality because of (26)

$$\|P_{\boldsymbol{\mu}_\tau} - P_{\boldsymbol{\nu}_\tau}\|_1 \geq \|\boldsymbol{\mu}_{\tau,uv} - \boldsymbol{\nu}_{\tau,uv}\|_1, \ \forall(u,v) \in E_\tau \ . \quad (28)$$

Adding them together gives

$$(2n_\tau - 1)\|P_{\boldsymbol{\mu}_\tau} - P_{\boldsymbol{\nu}_\tau}\|_1 \geq \|\boldsymbol{\mu}_\tau - \boldsymbol{\nu}_\tau\|_1 \geq \|\boldsymbol{\mu}_\tau - \boldsymbol{\nu}_\tau\|_2 \ . \quad (29)$$

According to Pinsker's inequality [3], we have

$$\begin{aligned} d_\phi(\boldsymbol{\mu}_\tau||\boldsymbol{\nu}_\tau) &= KL(P_{\boldsymbol{\mu}_\tau}, P_{\boldsymbol{\nu}_\tau}) \geq \frac{1}{2}\|P_{\boldsymbol{\mu}_\tau} - P_{\boldsymbol{\nu}_\tau}\|_1^2 \\ &\geq \frac{1}{2(2n_\tau - 1)^2}\|\boldsymbol{\mu}_\tau - \boldsymbol{\nu}_\tau\|_2^2 \ . \quad (30) \end{aligned}$$

Multiplying α on both sides and letting $\alpha \geq \beta(2n_\tau - 1)^2$ complete the proof. ∎

To prove the convergence of the objective function, we define a residual term R_τ^{t+1} as

$$R_\tau^{t+1} = \rho_\tau \langle \boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^*, \boldsymbol{\theta}_\tau \rangle \ , \quad (31)$$

where $\boldsymbol{\mu}_\tau^*$ is the optimal solution for tree τ. We show that R_τ^{t+1} satisfies the following inequality:

Lemma 2 *Let $\{\boldsymbol{m}_\tau, \boldsymbol{\mu}_\tau, \boldsymbol{\lambda}_\tau\}$ be the sequences generated by Bethe-ADMM. Assume $\alpha \geq \max_\tau\{\beta(2n_\tau - 1)^2\}$. For any $\boldsymbol{\mu}_\tau^* \in L(\tau)$, we have*

$$\begin{aligned} R_\tau^{t+1} &\leq \langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1} \rangle + \alpha\big(d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^t) - d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^{t+1})\big) \\ &+ \frac{\beta}{2}\big(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^t\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^t\|_2^2 - \|\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^t\|_2^2\big) \ , \quad (32) \end{aligned}$$

where R_τ^{t+1} is defined in (31).

Proof: Since $\boldsymbol{m}_\tau^{t+1}$ is the optimal solution for (21), for any $\boldsymbol{\mu}_\tau^* \in L(\tau)$, we have the following inequality:

$$\langle \mathbf{y}_r^t + \alpha(\nabla\phi(\boldsymbol{m}_\tau^{t+1}) - \nabla\phi(\boldsymbol{m}_\tau^t)), \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1} \rangle \geq 0 \ . \quad (33)$$

Substituting (16) into (33) and rearranging the terms, we have

$$\begin{aligned} R_\tau^{t+1} &\leq \langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1} \rangle + \beta\langle \boldsymbol{m}_\tau^t - \boldsymbol{\mu}_\tau^t, \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1} \rangle \\ &+ \alpha\langle \nabla\phi(\boldsymbol{m}_\tau^{t+1}) - \nabla\phi(\boldsymbol{m}_\tau^t), \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1} \rangle \ . \quad (34) \end{aligned}$$

The second term in the RHS of (34) is equivalent to

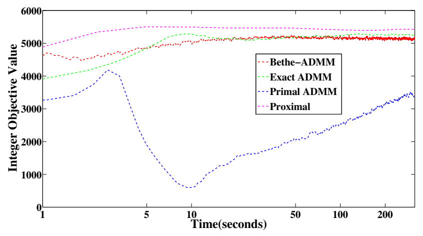
$$\begin{aligned} 2\langle \boldsymbol{m}_\tau^t - \boldsymbol{\mu}_\tau^t, \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1} \rangle &= \|\boldsymbol{m}_\tau^t - \boldsymbol{m}_\tau^{t+1}\|_2^2 \\ + \|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^t\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^t\|_2^2 &- \|\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^t\|_2^2. \quad (35) \end{aligned}$$

The third term in the RHS of (34) can be rewritten as

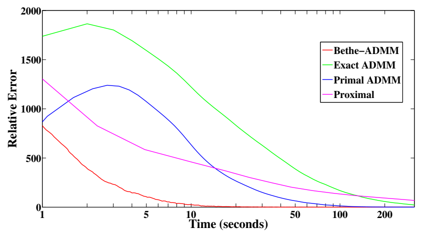
$$\begin{aligned} &\langle \nabla\phi(\boldsymbol{m}_\tau^{t+1}) - \nabla\phi(\boldsymbol{m}_\tau^t), \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1} \rangle \\ &= d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^t) - d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^{t+1}) - d_\phi(\boldsymbol{m}_\tau^{t+1}||\boldsymbol{m}_\tau^t). \quad (36) \end{aligned}$$

Substituting (35) and (36) into (34) and using Lemma 1 complete the proof. ∎

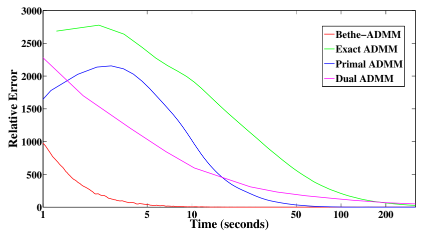
We next show that the first term in the RHS of (32) satisfies the following result:



(a) Rounded solution with $a = 0.5$.



(b) Relative error with $a = 0.5$.



(c) Relative error with $a = 1$.

Figure 1: Results of Bethe-ADMM, Exact ADMM, Primal ADMM and proximal algorithms on two simulation datasets. Figure 1(a) plots the value of the decoded integer solution as a function of runtime (seconds). Figure 1(b) and 1(c) plot the relative error with respect to the optimal LP objective as a function of runtime (seconds). For Bethe-ADMM, we set $\alpha = \beta = 0.05$. For Exact ADMM, we set $\beta = 0.05$. For Primal ADMM, we set $\beta = 0.5$. Bethe-ADMM converges faster than other primal based algorithms.

Lemma 3 *Let $\{\boldsymbol{m}_\tau, \boldsymbol{\mu}_\tau, \boldsymbol{\lambda}_\tau\}$ be the sequences generated by Bethe-ADMM. For any $\boldsymbol{\mu}^*_\tau \in L(\tau)$, we have*

$$\sum_{\tau=1}^{|\mathbb{T}|}\langle \boldsymbol{\lambda}^t_\tau, \boldsymbol{\mu}^*_\tau - \boldsymbol{m}^{t+1}_\tau\rangle \le \frac{1}{2\beta}(\|\boldsymbol{\lambda}^t_\tau\|_2^2 - \|\boldsymbol{\lambda}^{t+1}_\tau\|_2^2) + \frac{\beta}{2}\left(\|\boldsymbol{\mu}^*_\tau - \boldsymbol{m}^{t+1}_\tau\|_2^2 - \|\boldsymbol{\mu}^*_\tau - \boldsymbol{\mu}^{t+1}_\tau\|_2^2\right) .$$

Proof: Let μ_i be the ith component of $\boldsymbol{\mu}$. We augment $\boldsymbol{\mu}_\tau, \boldsymbol{m}_\tau$ and $\boldsymbol{\lambda}_\tau$ in the following way: If $\boldsymbol{\mu}_i$ is not a component of $\boldsymbol{\mu}_\tau$, we set $\boldsymbol{\mu}_{\tau,i} = 0, \boldsymbol{m}_{\tau,i} = 0$ and $\boldsymbol{\lambda}_{\tau,i} = 0$; otherwise, they are the corresponding components from $\boldsymbol{\mu}_\tau, \boldsymbol{m}_\tau$ and $\boldsymbol{\lambda}_\tau$ respectively. We can then rewrite (22) in the following equivalent component-wise form:

$$\mu_i^{t+1} = \underset{\mu_i}{\operatorname{argmin}} \sum_{\tau=1}^{|\mathbb{T}|}\left(\langle \lambda^t_{\tau,i}, m^{t+1}_{\tau,i} - \mu_{\tau,i}\rangle + \frac{\beta}{2}\|m^{t+1}_{\tau,i} - \mu_{\tau,i}\|_2^2\right) .$$

For any $\boldsymbol{\mu}^*_\tau \in L(\tau)$, we have the following optimality condition:

$$-\sum_{\tau=1}^{|\mathbb{T}|}\langle \lambda^t_{\tau,i} + \beta(m^{t+1}_{\tau,i} - \mu^{t+1}_{\tau,i}), \mu^*_{\tau,i} - \mu^{t+1}_{\tau,i}\rangle \ge 0 . \quad (37)$$

Combining all the components of $\boldsymbol{\mu}^{t+1}$, we can rewrite (37) in the following vector form:

$$-\sum_{\tau=1}^{|\mathbb{T}|}\langle \boldsymbol{\lambda}^t_\tau + \beta(\boldsymbol{m}^{t+1}_\tau - \boldsymbol{\mu}^{t+1}_\tau), \boldsymbol{\mu}^*_\tau - \boldsymbol{\mu}^{t+1}_\tau\rangle \ge 0 . \quad (38)$$

Rearranging the terms yields

$$\begin{aligned}
&\sum_{\tau=1}^{|\mathbb{T}|}\langle \boldsymbol{\lambda}^t_\tau, \boldsymbol{\mu}^*_\tau - \boldsymbol{m}^{t+1}_\tau\rangle \\
&\le \sum_{\tau=1}^{|\mathbb{T}|}\langle \boldsymbol{\lambda}^t_\tau, \boldsymbol{\mu}^{t+1}_\tau - \boldsymbol{m}^{t+1}_\tau\rangle - \sum_{\tau=1}^{|\mathbb{T}|}\beta\langle \boldsymbol{m}^{t+1}_\tau - \boldsymbol{\mu}^{t+1}_\tau, \boldsymbol{\mu}^*_\tau - \boldsymbol{\mu}^{t+1}_\tau\rangle \\
&= \sum_{\tau=1}^{|\mathbb{T}|}\langle \boldsymbol{\lambda}^t_\tau, \boldsymbol{\mu}^{t+1}_\tau - \boldsymbol{m}^{t+1}_\tau\rangle + \frac{\beta}{2}\sum_{\tau=1}^{|\mathbb{T}|}\left(\|\boldsymbol{\mu}^*_\tau - \boldsymbol{m}^{t+1}_\tau\|_2^2 - \|\boldsymbol{\mu}^*_\tau - \boldsymbol{\mu}^{t+1}_\tau\|_2^2 - \|\boldsymbol{\mu}^{t+1}_\tau - \boldsymbol{m}^{t+1}_\tau\|_2^2\right) . \quad (39)
\end{aligned}$$

Recall $\boldsymbol{\mu}^{t+1}_\tau - \boldsymbol{m}^{t+1}_\tau = \frac{1}{\beta}(\boldsymbol{\lambda}^t_\tau - \boldsymbol{\lambda}^{t+1}_\tau)$ in (23), then

$$\langle \boldsymbol{\lambda}^t_\tau, \boldsymbol{\mu}^{t+1}_\tau - \boldsymbol{m}^{t+1}_\tau\rangle - \frac{\beta}{2}\|\boldsymbol{\mu}^{t+1}_\tau - \boldsymbol{m}^{t+1}_\tau\|_2^2 = \frac{1}{2\beta}(\|\boldsymbol{\lambda}^t_\tau\|_2^2 - \|\boldsymbol{\lambda}^{t+1}_\tau\|_2^2) . \quad (40)$$

Plugging (40) into (39) completes the proof. ■

We also need the following lemma which can be found in [6]. We omit the proof due to lack of space.

Lemma 4 *Let $\{\boldsymbol{m}_\tau, \boldsymbol{\mu}_\tau, \boldsymbol{\lambda}_\tau\}$ be the sequences generated by Bethe-ADMM. Then*

$$\sum_{\tau=1}^{|\mathbb{T}|}\|\boldsymbol{m}^{t+1}_\tau - \boldsymbol{\mu}^t_\tau\|_2^2 \ge \sum_{\tau=1}^{|\mathbb{T}|}\|\boldsymbol{m}^{t+1}_\tau - \boldsymbol{\mu}^{t+1}_\tau\|_2^2 + \|\boldsymbol{\mu}^{t+1}_\tau - \boldsymbol{\mu}^t_\tau\|_2^2 .$$

Theorem 1 *Assume the following hold: (1) $\boldsymbol{m}^0_\tau$ and $\boldsymbol{\mu}^0_\tau$ are uniform tree-structured distributions, $\forall \tau = 1, \dots, |\mathbb{T}|$ (2) $\boldsymbol{\lambda}^0_\tau = \mathbf{0}, \forall \tau = 1, \dots, |\mathbb{T}|$; (3) $\max_\tau d_\phi(\boldsymbol{\mu}^*_\tau\|\boldsymbol{m}^0_\tau) = D_{\boldsymbol{\mu}}$; (4) $\alpha \ge \max_\tau\{\beta(2n_\tau - 1)^2\}$ holds. Denote $\bar{\boldsymbol{m}}^T_\tau = \frac{1}{T}\sum_{t=0}^{T-1}\boldsymbol{m}^t_\tau$ and $\bar{\boldsymbol{\mu}}^T_\tau = \frac{1}{T}\sum_{t=0}^{T-1}\boldsymbol{\mu}^t_\tau$. For any T and the optimal solution $\boldsymbol{\mu}^*$, we have*

$$\sum_{\tau=1}^{|\mathbb{T}|}\left(\rho_\tau\langle \bar{\boldsymbol{m}}^T_\tau - \boldsymbol{\mu}^*_\tau, \theta_\tau\rangle + \frac{\beta}{2}\|\bar{\boldsymbol{m}}^T_\tau - \bar{\boldsymbol{\mu}}^T_\tau\|_2^2\right) \le \frac{D_{\boldsymbol{\mu}}\alpha|\mathbb{T}|}{T} .$$

Proof: Summing (32) over τ from 1 to $|\mathbb{T}|$ and using Lemma 3, we have:

$$\begin{aligned}
&\sum_{\tau=1}^{|\mathbb{T}|}\left(R^{t+1}_\tau + \frac{\beta}{2}\|\boldsymbol{m}^{t+1}_\tau - \boldsymbol{\mu}^t_\tau\|_2^2\right) \\
&\le \sum_{\tau=1}^{|\mathbb{T}|}\frac{1}{2\beta}(\|\boldsymbol{\lambda}^t_\tau\|_2^2 - \|\boldsymbol{\lambda}^{t+1}_\tau\|_2^2) + \frac{\beta}{2}(\|\boldsymbol{\mu}^*_\tau - \boldsymbol{\mu}^t_\tau\|_2^2 - \|\boldsymbol{\mu}^*_\tau - \boldsymbol{\mu}^{t+1}_\tau\|_2^2) \\
&+ \frac{\beta}{2}\left(\|\boldsymbol{\mu}^*_\tau - \boldsymbol{m}^{t+1}_\tau\|_2^2 - \|\boldsymbol{\mu}^*_\tau - \boldsymbol{m}^t_\tau\|_2^2\right) \\
&+ \alpha\left(d_\phi(\boldsymbol{\mu}^*_\tau\|\boldsymbol{m}^t_\tau) - d_\phi(\boldsymbol{\mu}^*_\tau\|\boldsymbol{m}^{t+1}_\tau)\right) . \quad (41)
\end{aligned}$$

Summing over the above from $t = 0$ to $T-1$, we have

$$\begin{aligned}
&\sum_{t=0}^{T-1}\sum_{\tau=1}^{|\mathbb{T}|}\left(R_\tau^{t+1} + \frac{\beta}{2}\|\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^t\|_2^2\right) \\
&\leq \sum_{\tau=1}^{|\mathbb{T}|}\frac{1}{2\beta}(\|\boldsymbol{\lambda}_\tau^0\|_2^2 - \|\boldsymbol{\lambda}_\tau^T\|_2^2) + \frac{\beta}{2}\left(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^0\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^T\|_2^2\right) \\
&+ \frac{\beta}{2}\left(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^T\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^0\|_2^2\right) \\
&+ \alpha\left(d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^0) - d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^T)\right) \\
&\leq \sum_{\tau=1}^{|\mathbb{T}|}\frac{\beta}{2}\|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^T\|_2^2 + \alpha\left(d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^0) - d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^T)\right) \\
&\leq \sum_{\tau=1}^{|\mathbb{T}|}\alpha d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^0)\ , \qquad (42)
\end{aligned}$$

where we use Lemma 1 to derive (42). Applying Lemma 4 and Jensen's inequality yield the desired bound. ■

Theorem 1 establishes the $O(1/T)$ convergence rate for the Bethe-ADMM in ergodic sense. As $T \to \infty$, the objective value $\sum_{\tau=1}^{|\mathbb{T}|}\rho_\tau\langle\bar{\boldsymbol{m}}_\tau^T, \theta_\tau\rangle$ converges to the optimal value and the equality constraints are also satisfied.

3.4 Extension to MRFs with General Factors

Although we present Bethe-ADMM in the context of pairwise MRFs, it can be easily generalized to handle MRFs with general factors. For a general MRF, we can view the dependency graph as a factor graph [12], a bipartite graph $G = (V \cup F, E)$, where V and F are disjoint set of variable nodes and factor nodes and E is a set of edges, each connecting a variable node and a factor node. The distribution $P(\boldsymbol{x})$ takes the form: $P(\boldsymbol{x}) \propto \exp\left\{\sum_{u\in V} f_u(x_u) + \sum_{\alpha\in F} f_\alpha(\boldsymbol{x}_\alpha)\right\}$. The relaxed LP for general MRFs can be constructed in a similar fashion with that for pairwise MRFs.

We can then decompose the relaxed LP to subproblems defined on factor trees and impose equality constraints to enforce consistency on the shared variables among the subproblems. Each subproblem can be solved efficiently using the sum-product algorithm for factor trees and the Bethe-ADMM algorithm for general MRFs bears similar structure with that for pairwise MRFs.

4 Experimental Results

We compare the Bethe-ADMM algorithm with several other state-of-the-art MAP inference algorithms. We show the comparison results with primal based MAP inference algorithms in Section 4.1 and dual based MAP inference algorithm in Section 4.2 respectively. We also show in Section 4.3 how tree decomposition benefits the performance of Bethe-ADMM. We run experiments in Section 4.1-4.3 using sequential updates. To illustrate the scalability of our algorithm, we run parallel Bethe-ADMM on a multicore machine and show the linear speedup in Section 4.4.

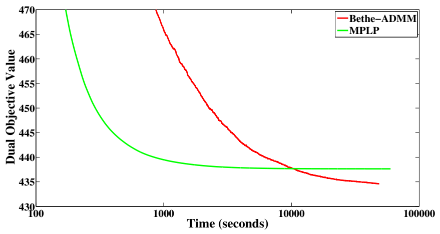


Figure 2: Both Bethe-ADMM and MPLP are run for sufficiently long, i.e., 50000 iterations. The dual objective value is plotted as a function of runtime (seconds). The MPLP algorithm gets stuck and does not reach the global optimum.

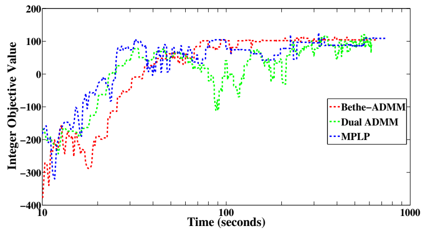
4.1 Comparison with Primal based Algorithms

We compare the Bethe-ADMM algorithm with the proximal algorithm [18], Exact ADMM algorithm and Primal ADMM algorithm [15]. For the proximal algorithm, we choose the Bregman divergence as the sum of KL-divergences across all node and edge distributions. Following the methodology in [18], we terminate the inner loop if the maximum constraint violation of $L(G)$ is less than 10^{-3} and set $w^t = t$. Similarly, in applying the Exact ADMM algorithm, we terminate the loop for solving $\boldsymbol{m}_\tau$ if the maximum constraint violation of $L(\tau)$ is less than 10^{-3}. For the Exact ADMM and Bethe-ADMM algorithm, we use 'edge decomposition': each τ is simply an edge of the graph and $|\mathbb{T}| = |E|$. To obtain the integer solution, we use node-based rounding: $x_u^* = \operatorname{argmax}_{x_u} \mu_u(x_u)$.

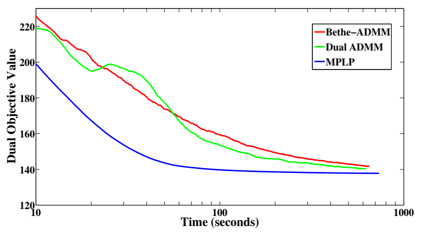
We show experimental results on two synthetic datasets. The underlying graph of each dataset is a three dimensional $m \times n \times t$ grid. We generate the potentials as follows: We set the nodewise potentials as random numbers from $[-a, a]$, where $a > 0$. We set the edgewise potentials according to the Potts model, i.e., $\theta_{uv}(x_u, x_v) = b_{uv}$ if $x_u = x_v$ and 0 otherwise. We choose b_{uv} randomly from $[-1, 1]$. The edgewise potentials penalize disagreement if $b_{uv} > 0$ and penalize agreement if $b_{uv} < 0$. We generate datasets using $m = 20, n = 20, t = 16, k = 6$ with varying a.

Figure 1(a) shows the plots of (1) on one synthetic dataset and we find that the algorithms have similar performances on other simulation datasets. We observe that all algorithms converge to the optimal value $\langle\boldsymbol{\mu}^*, \boldsymbol{f}\rangle$ of (3) and we plot the relative error with respect to the optimal value $|\langle\boldsymbol{\mu}^* - \boldsymbol{\mu}_t, \boldsymbol{f}\rangle|$ on the two datasets in Figure 1(b) and 1(c).

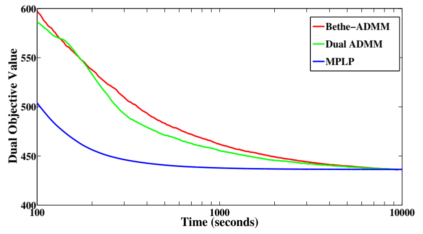
Overall, the Bethe-ADMM algorithm converges faster than other primal algorithms. We observe that the proximal algorithm and Exact ADMM algorithm are the slowest, due to the sequential projection step. In terms of the decoded integer solution, the Bethe-ADMM, Exact ADMM and proximal algorithm have similar performances. We



(a) Rounded integer solution on 1jo8.



(b) Dual value on 1jo8.



(c) Dual value on 1or7.

Figure 3: Results of Bethe-ADMM, MPLP and Dual ADMM algorithms on two protein design datasets. Figure 3(a) plots the the value of the decoded integer solution as a function of runtime (seconds). Figure 3(b) and 3(c) plot the dual value as a function of runtime (seconds). For Dual ADMM, we set $\beta = 0.05$. For Bethe-ADMM, we set $\alpha = \beta = 0.1$. Bethe-ADMM and Dual ADMM have similar performance in terms of convergence. All three methods have comparable performances for the decoded integer solution.

also note that a higher objective function value does not necessarily lead to a better decoded integer solution.

4.2 Comparison with Dual based Algorithms

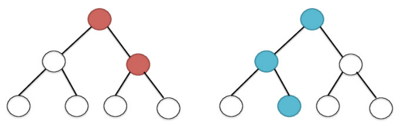
In this section, we compare the Bethe-ADMM algorithm with the MPLP algorithm [7] and the Dual ADMM algorithm [15]. We conduct experiments on protein design problems [26]. In these problems, we are given a 3D structure and the goal is to find a sequence of amino-acids that is the most stable for that structure. The problems are modeled by nodewise and pairwise factors and can be posed as finding a MAP assignment for the given model. This is a demanding setting in which each problem may have hundreds of variables with 100 possible states on average.

We run the algorithms on two problems with different sizes [26], i.e., 1jo8 (58 nodes and 981 edges) and 1or7 (180 nodes and 3005 edges). For the MPLP and Dual ADMM algorithm, we plot the value of the integer programming problem (1) and its dual.. For Bethe-ADMM algorithm, we plot the value of dual LP of (3) and the integer programming problem (1). Note that although Bethe-ADMM and Dual ADMM have different duals, their optimal values are the same. We run the Bethe-ADMM based on edge decomposition. Figure 3 shows the result.

We observe that the MPLP algorithm usually converges faster, but since it is a coordinate ascent algorithm, it can stop prematurely and yield suboptimal solutions. Figure 2 shows that on the 1fpo dataset, the MPLP algorithm converges to a suboptimal solution. We note that the convergence time of the Bethe-ADM and Dual ADM are similar. The three algorithms have similar performance in terms of the decoded integer solution.

4.3 Edge based vs Tree based

In the previous experiments, we use 'edge decomposition' for the Bethe-ADMM algorithm. Since our algorithm can work for any tree-structured graph decomposition, we want to empirically study how the decomposition affects the performance of the Bethe-ADMM algorithm. In the following experiments, we show that if we can utilize the graph


(a)

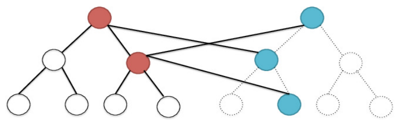
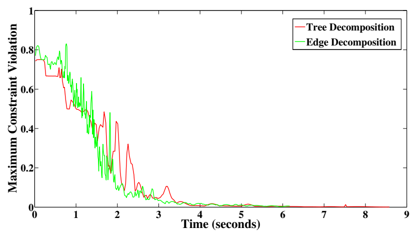

(b)

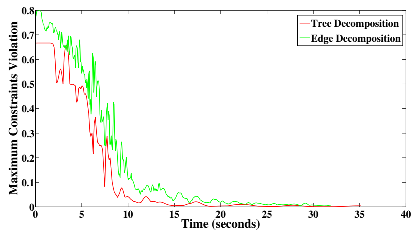
Figure 4: A simulation dataset with $m = 2$, $s = 7$ and $n = 3$. In 4(a), the red nodes (S_{12}) are sampled from tree 1 and the blue nodes (D_{12}) are sampled from tree 2. In 4(b) , sampled nodes are connected by cross-tree edges (E_{12}). Tree 1 with nodes in D_{12} and edges in E_{12} still form a tree, denoted by solid lines. This augmented tree is a tree-structured subgraph for Bethe-ADMM.

structure when decomposing the graph, the Bethe-ADMM algorithm will have better performance compared to simply using 'edge decomposition', which does not take the graph structure into account.

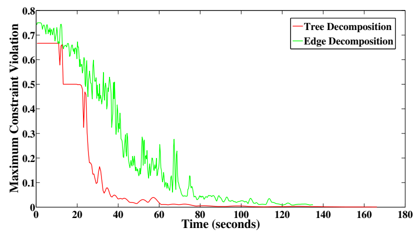
We conduct experiments on synthetic datasets. We generate MRFs whose dependency graphs consist of several tree-structured graphs and cross-tree edges to introduce cycles. To be more specific, we first generate m binary tree structured MRFs each with s nodes. Then for each ordered pair of tree-structured MRFs $(i, j), 1 \leq i, j \leq m, i \neq j$, we uniformly sample n nodes from MRF i with replacement and uniformly sample n $(n \leq s)$ nodes from MRF j without replacement, resulting in two node sets S_{ij} and D_{ij}. We then connect the nodes in S_{ij} and D_{ij}, denoting them as E_{ij}. We repeat this process for every pair of trees. By construction, the graph consisting of tree i, nodes in D_{ij} and edges in E_{ij}, $\forall j \neq i$ is still a tree. We will use these m augmented trees as the tree-structured subgraphs for the Bethe-ADMM algorithm. Figure 4 illustrates the graph generation and tree decomposition process. A simple calculation shows that for this particular tree decomposition, $O(m^2nk)$ equality constraints are maintained, while for edge decomposition, $O(msk + m^2nk)$ are maintained. When the graph has dominant tree structure, tree decomposition leads to much less number of equality constraints.



(a) $s = 1023$.



(b) $s = 4095$.



(c) $s = 16383$.

Figure 5: Results of Bethe-ADMM algorithms based on tree and edge decomposition on three simulation datasets with $m = 10, n = 20$. The maximum constraint violation in $L(G)$ is plotted as a function of runtime (seconds). For both algorithms, we set $\alpha = \beta = 0.05$. The tree based Bethe-ADMM algorithm has better performance than that of the edge based Bethe-ADMM when the tree structure is more dominant in G.

For the experiments, we run the Bethe-ADMM algorithm based on tree and edge decomposition with different values of s, keeping m and n fixed. It is easy to see that the tree structure becomes more dominant when s becomes larger. Since we observe that both algorithms first converge to the optimal value of (3) and then the equality constraints are gradually satisfied, we evaluate the performance by computing the maximum constraint violation of $L(G)$ at each iteration for both algorithms. The faster the constraints are satisfied, the better the algorithm is. The results are shown in Figure 5. When the tree structure is not obvious, the two algorithms have similar performances. As we increase s and the tree structure becomes more dominant, the difference between the two algorithms is more pronounced. We attribute the superior performance to the fact that for the tree decomposition case, much fewer number of equality constraints are imposed and each subproblem on tree can be solved efficiently using the sum-product algorithm.

4.4 Scalability Experiments on Multicores

The dataset used in this section is the Climate Research Unit (CRU) precipitation dataset [16], which has monthly precipitation from the years 1901-2006. The dataset is of high gridded spatial resolution (360 $\times$ 720, i.e., 0.5 degree latitude $\times$ 0.5 degree longitude) and includes the precipitation over land.

Our goal is to detect major droughts based on precipitation. We formulate the problem as the one of estimating the most likely configuration of a binary MRF, where each node represents a location. The underlying graph is a three dimensional grid (360 $\times$ 720 $\times$ 106) with 7,146,520 nodes and each node can be in two possible states: dry and normal. We run the Bethe-ADMM algorithm on the CRU dataset and detect droughts based on the integer solution after node-based rounding. For the details of the this experiment, we refer to readers to [5]. Our algorithm successfully detects nearly all the major droughts of the last century. We also examine how the Bethe-ADMM algorithm scales on the CRU dataset with more than 7 million variables. We run the Open MPI code with different number of cores and the result in Figure 6 shows that we obtain almost linear speedup with the number of cores.

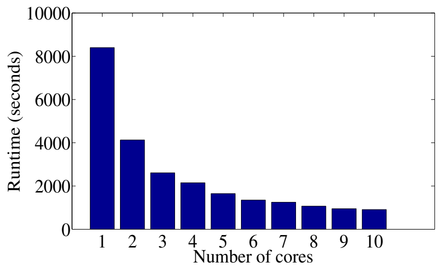


Figure 6: The Open MPI implementation of Bethe-ADMM has almost linear speedup on the CRU dataset with more than 7 million nodes.

5 Conclusions

We propose a provably convergent MAP inference algorithm for large scale MRFs. The algorithm is based on the 'tree decomposition' idea from the MAP inference literature and the alternating direction method from the optimization literature. Our algorithm solves the tree structured subproblems efficiently via the sum-product algorithm and is inherently parallel. The empirical results show that the new algorithm, in its sequential version, compares favorably to other existing approximate MAP inference algorithm in terms of running time and accuracy. The experimental results on large datasets demonstrate that the parallel version scales almost linearly with the number of cores in the multi-core setting.

Acknowledgements

This research was supported in part by NSF CAREER Grant IIS-0953274, and NSF Grants IIS-1029711, IIS-0916750, and IIS-0812183. The authors would like to thank Stefan Liess and Peter K. Snyder for the helpful discussion on the CRU experiment. The authors are grateful for the technical support from the University of Minnesota Supercomputing Institute (MSI).

References

[1] 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.

[2] Y. Censor and S. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1998.

[3] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[4] C. Chekuri, S. Khanna, J. Naor, and L. Zosin. A linear programming formulation and approximation algorithms for the metric labeling problem. *SIAM Journal on Discrete Mathematics*, 18(3):608–625, Mar. 2005.

[5] Q. Fu, A. Banerjee, S. Liess, and P. K. Snyder. Drought detection of the last century: An MRF-based approach. In *Proceedings of the SIAM International Conference on Data Mining*, 2012.

[6] Q. Fu, H. Wang, A. Banerjee, S. Liess, and P. K. Snyder. MAP inference on million node graphical models: KL-divergence based alternating directions method. Technical report, Computer Science and Engineering Department, University of Minesota, 2012.

[7] A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*, 2007.

[8] B. He and X. Yuan. On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2):700–709, 2012.

[9] V. Jojic, S. Gould, and D. Koller. Fast and smooth: Accelerated dual decomposition for MAP inference. In *Proceedings of the twenty-Seventh International Conference on Machine Learning*, 2010.

[10] S. P. Kasiviswanathan, P. Melville, A. Banerjee, and V. Sindhwani. Emerging topic detection using dictionary learning. In *Proceedings of the Twentieth ACM international conference on Information and knowledge management*, 2011.

[11] N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):531 –552, march 2011.

[12] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[13] A. F. Martins. *The Geometry of Constrained Structured Prediction: Applications to Inference and Learning of Natural Language Syntax*. PhD thesis, Carnegie Mellon University, 2012.

[14] A. F. Martins, P. M. Aguiar, M. A. Figueiredo, N. A. Smith, and E. P. Xing. An augmented Lagrangian approach to constrained MAP inference. In *Proceedings of the Twenty-Eighth International Conference on Machine Learning*, 2011.

[15] O. Meshi and A. Globerson. An alternating direction method for dual MAP LP relaxation. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2011.

[16] T. D. Mitchell, T. R. Carter, P. D. Jones, M. Hulme, and M. New. *A comprehensive set of high-resolution grids of monthly climate for Europe and the globe: the observed record (1901-2000) and 16 scenarios (2001-2100)*. Tyndall Centre for Climate Change Research, 2004.

[17] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[18] P. Ravikumar, A. Agarwal, and M. J. Wainwright. Message-passing for graph-structured linear programs: Proximal methods and rounding schemes. *Journal of Machine Learning Research*, 11:1043–1080, 2010.

[19] 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*. MIT Press, 2011.

[20] D. Sontag and T. Jaakkola. Tree block coordinate descent for MAP in graphical models. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*.

[21] D. Tarlow, D. Batra, P. Kohli, and V. Kolmogorov. Dynamic tree block coordinate ascent. In *Proceedings of the Twenty-Eighth International Conference on Machine Learning*, 2011.

[22] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Transactions of Information Theory*, 51(11):3697–3717, 2005.

[23] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

[24] H. Wang and A. Banerjee. Online alternating direction method. In *Proceedings of the Twenty-Ninth International Conference on Machine Learning*, 2012.

[25] J. Yang and Y. Zhang. Alternating direction algorithms for l1-problems in compressive sensing. *SIAM Journal on Scientific Computing*, 33(1):250–278, 2011.

[26] C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation: an empirical study. *Journal of Machine Learning Research*, 7:1887–1907, 2006.

Building Bridges: Viewing Active Learning from the Multi-Armed Bandit Lens

Ravi Ganti and Alexander G. Gray
Computational Science and Engineering.
Georgia Institute of Technology
Atlanta, GA 30332
gmravi2003@gatech.edu, agray@cc.gatech.edu

Abstract

In this paper we propose a multi-armed bandit inspired, pool based active learning algorithm for the problem of binary classification. By carefully constructing an analogy between active learning and multi-armed bandits, we utilize ideas such as lower confidence bounds, and self-concordant regularization from the multi-armed bandit literature to design our proposed algorithm. Our algorithm is a sequential algorithm, which in each round assigns a sampling distribution on the pool, samples one point from this distribution, and queries the oracle for the label of this sampled point. The design of this sampling distribution is also inspired by the analogy between active learning and multi-armed bandits. We show how to derive lower confidence bounds required by our algorithm. Experimental comparisons to previously proposed active learning algorithms show superior performance on some standard UCI data-sets.

1 Introduction

In the classical passive binary classification problem one has access to labeled samples $\mathcal{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, drawn from an unknown distribution P defined on a domain $\mathcal{X} \times \{-1, +1\}$, where $\mathcal{X} \subset \mathbb{R}^d$. The points $\{x_1, \ldots, x_n\}$ are sampled i.i.d. from the marginal distribution $P_{\mathcal{X}}$, and the labels $y_1, \ldots, y_n$ are sampled from the conditional distribution $P_{Y|X=x}$. Classical learning algorithms, such as boosting, SVMs, logistic regression choose a hypothesis class $\mathcal{H}$, and an appropriate loss function $L(\cdot)$, and solve some sort of an empirical risk minimization problem to return a hypothesis $\hat{h} \in \mathcal{H}$, whose risk $R(h) \overset{\text{def}}{=} \mathbb{E}_{x,y\sim P} L(yh(x))$ is small. However, in domains such as speech recognition, natural language processing, there is generally a lacuna of labeled data, and obtaining labels for unlabeled data is both tedious, and expensive. In such cases it is of both theoretical and practical interest to design learning algorithms, which need only a few labeled examples for training, and also guarantee good performance on unseen data. In recent years active learning (AL) has emerged as a very popular framework for solving machine learning problems with limited labeled data (Settles, 2009). In this framework the learning algorithm is "active", and is allowed to query, an oracle $\mathcal{O}$, for the label of those points which it feels are maximally informative for the learning process. The hope is that by using few, but wisely chosen labels the active learning algorithm will be able to learn as well as a passive learning algorithm, which has access to lots of labeled data.

Broadly speaking AL algorithms can be classified into three kinds, namely membership query (MQ) based algorithms, stream based algorithms and pool based algorithms. All these three kinds of AL algorithms query $\mathcal{O}$ for the label of the point, but differ from each other in the nature of the queries. In MQ based algorithms the active learner can query $\mathcal{O}$, for the label of a point in the input space $\mathcal{X}$. However, this query need not necessarily be from the support of the marginal distribution $\mathcal{P}_{\mathcal{X}}$. MQ algorithms might work poorly when the oracle $\mathcal{O}$ is a human annotator (Baum and Lang, 1992). Stream based AL algorithms (Cohn et al., 1994; Chu et al., 2011) sample a point x from the marginal distribution $\mathcal{P}_{\mathcal{X}}$, and decide on-the-fly whether to query $\mathcal{O}$ for the label of x. Stream based AL algorithms are computationally efficient, and most appropriate when the underlying distribution changes with time. In pool based AL we are provided with a pool $\mathcal{P} = \{x_1, \ldots x_n\}$ of unlabeled points, which have been sampled i.i.d from the marginal distribution $P_{\mathcal{X}}$, and a labeling oracle $\mathcal{O}$, which when queried for the label of x, returns $y \sim P_{Y|X=x}$. Algorithms in the pool based setting have the luxury of deciding which points to query by looking at the entire pool.

Contributions. In this paper we shall deal with pool based active learning. We shall also assume that we have a query budget, B, of the maximum number of queries the AL algorithm can issue to the oracle.

We view AL from the lens of exploration-exploitation trade-off. The concept of exploration-exploitation is central to problems in decision making under uncertainty, and is best illustrated by the multi-armed bandit (MAB) problem [1] [2]. The MAB problem is a B round game, where in a generic round t, the player has to pull one among k arms of a multi-armed bandit. On doing so the player suffers a loss L_t. The player does not get to know the loss he could have suffered if he had pulled a different arm. The goal of the player is to minimize the cumulative loss suffered over B rounds. In each round the player needs to resolve the dilemma of whether to explore an arm which has not been pulled in the past, or whether to exploit the knowledge of the cumulative losses of the arms that have been pulled in the past. We provide a pool based, sequential AL algorithm called LCB-AL, which is motivated by applying algorithmic ideas from the problem of multi-armed bandits to the problem of AL. In order to do so we build a bridge between the MAB problem and AL problem, providing an equivalence between the arms of a MAB problem, and the hypothesis in $\mathcal{H}$, and mitigating the problem of absence of an explicit loss signal in AL, unlike MAB. Establishing this analogy is not very straightforward, but once done allows us to readily use tools such as lower confidence bounds (Auer et al., 2002a), and self-concordant regularization (Abernethy et al., 2008) in the design of LCB-AL. To our knowledge, our work is one of the first in trying to use bandit type ideas for active learning, and we strongly believe that one can build extremely practical, yet very simple and scalable algorithms by understanding the interplay between multi-armed bandits and active learning.

In section 2 we take the first steps towards building an analogy between MAB and AL. This inspires us to use a very successful algorithm from the MAB literature, for the problem of AL. We build the technical tools needed to fill in the details of our proposed algorithm in section 3. Section 4 discusses related work, and section 5 compares LCB-AL with other active learning algorithms on various datasets.

[1]Usually in the literature on MAB it is common to talk of rewards. For our purposes, it would be more convenient to talk of losses rather than rewards.

[2]We shall consider the non-stochastic bandit problem

2 Towards an analogy between Multi-armed bandits and Active Learning

In active learning the goal is to find a hypothesis $h \in \mathcal{H}$ with low risk, by using as little labeled data as possible. In other words, we want to quickly estimate the risk of different hypothesis, and discard suboptimal hypothesis. In MAB, the goal of the player is to design a strategy, that minimize the cumulative loss suffered by the player over T rounds. If the player knew the arms with the smallest possible cumulative loss then the optimal strategy would be to pull this arm in each and every round. Hence, in MAB the player wants to quickly detect the (near) optimal arm to pull. Looking from the lens of MAB, it is now natural to think of AL problem as a MAB problem, where the arms of the MAB are the different hypothesis in $\mathcal{H}$. While this is a satisfying connection there are two issues that still need to be resolved. 1) In the MAB problem, in each round we pull an arm of the MAB. If arms of the MAB were equivalent to the different hypothesis in $\mathcal{H}$, then how do we decide which "hypothesis to pull". 2) In MAB the player gets to see an explicit loss signal at the end of each round. However, in AL there is no such explicit loss signal, instead the feedback that is received is the label of the queried point x. Hence, the next question that arises is how could one use the label information as some kind of a loss signal? The following subsections attempt to resolve these issues.

2.1 Which hypothesis to pull?

A very popular approach in MAB to mitigate the exploration-exploitation trade-off is via the use of lower confidence bounds (LCB) (Auer et al., 2002b,a; Audibert et al., 2009a; Bubeck and Cesa-Bianchi, 2012) [3]. In the LCB approach, at the end of round t, for each arm a in the set of arms, we build a lower confidence bound, $\mathrm{LCB}_t(a)$ for the cumulative loss the player would have suffered, in hindsight, had he pulled arm a for the first t rounds. The choice of arm a_{t+1} to be pulled in the next round, i.e. round $t+1$ is the solution to the optimization problem $a_{t+1} \in \arg\min \mathrm{LCB}_t(a)$. Such lower confidence bounds can be derived via concentration inequalities (Auer et al., 2002b; Audibert et al., 2009b), and are generally expressed as $LCB_t(a) \stackrel{\text{def}}{=} \hat{L}_t(a) - U(\hat{L}_t(a))$, where $\hat{L}_t(a)$ is an estimate of the cumulative loss of arm a, the player would have suffered had he pulled a each time for the first t rounds, and $U(\hat{L}_t(a))$ is some measure of uncertainty (typically variance) of the cumulative

[3]Traditionally it has been called as the upper-confidence bound algorithm. Since, we are dealing with losses and not rewards, it is useful for our purpose, to rename this as LCB

loss of a, at the end of round t. The reason behind the success of confidence bounds in the MAB problem can be explained by the fact that $\mathrm{LCB}_t(a)$ captures both the knowledge of the cumulative loss, via $\hat{L}_t(a)$, as well as the uncertainty in this estimate, via $U(\hat{L}_t(a_t))$. By pulling the arm a_{t+1} in round $t+1$ of our MAB algorithm, and by updating our estimate of the cumulative loss of arm a_{t+1}, our updated estimate $\hat{L}_{t+1}(a_{t+1})$ is a better estimator as $U(\hat{L}_{t+1}(a_{t+1}))$ is potentially smaller than $U(\hat{L}_t(a_{t+1}))$.

One could use a similar technique even in AL. If one had some kind of a LCB on the risk of each hypothesis, then we could equate pulling a hypothesis as solving the optimization problem $h_{t+1} \in \arg\min_{h\in\mathcal{H}} \mathrm{LCB}_t(h)$, where $\mathrm{LCB}_t(h)$ is the lower confidence bound on the risk of $h \in \mathcal{H}$. An LCB for $R(h)$ can be obtained by utilizing the labeled data gathered over the run of the algorithm.

2.2 Absence of a loss signal in AL

When an arm is pulled in the MAB setting, the player suffers a loss, and this loss is used to update the LCB of the chosen arm. However, in AL there is no such explicit loss signal. One might come up with a proxy loss signal for the active learning problem which can then be used to update the lower confidence bound of all the hypothesis in $\mathcal{H}$. However, we take a different approach. The utility of the loss signal when the arm a_t is pulled in round t of the MAB problem is two folds. Firstly to update the cumulative loss of a_t, and secondly to decrease the uncertainty in the estimate of the cumulative loss of a_t. In AL when a certain point x is queried for its label, then this label information can be utilized to improve the error estimate of h_t as well as other hypothesis. Hence, it makes sense to query $\mathcal{O}$, for the label of some point x in P, such that its label information maximally reduces the variance of the estimate of risk of h_t. Hence, by conceptually viewing label information as a mechanism to reduce the variance of the risk estimate of different hypothesis, we have a disciplined way of deciding which points to query. Table 1 summarizes the analogy between AL and MAB.

2.3 Rough outline of LCB-AL.

LCB-AL is a sequential algorithm where in each round a probability distribution is placed on $\mathcal{P}$, and a single point is sampled from $\mathcal{P}$. We additionally assume that the hypothesis class $\mathcal{H}$ is convex. For the sake of simplicity we shall allow re-querying points in the pool, i.e. if a point x_i was first queried in some round t_1, then it might once again be queried in round t_2. If such a re-querying happens then we use the label that was returned by $\mathcal{O}$ in round t_1. [4] In order to construct lower confidence bounds on the risk of h we use importance weighting along with Bernstein type inequalities for martingales. The advantage of using importance weights, is that they facilitate data reuse, when an actively sampled data with one hypothesis class, is used in the future to learn a model from a different hypothesis class. The problem with such importance weighted estimators is that they have very high variance. The seminal work of Abernethy et al. (2008) showed that one could tackle the high variance of the importance weighted estimators, via the use of self-concordant barriers (Nesterov and Nemirovsky, 1994). This inspires us to use the self-concordant barrier of $\mathcal{H}$ along with lower confidence bounds in our algorithm. As a result in each round (see step 14 of algorithm 1) we solve the optimization problem

$$h_{t+1} \in \arg\min_{h\in\mathcal{H}} LCB_t(h) + \mathcal{R}(h),$$

where $\mathcal{R}(h)$ is the self-concordant barrier of $\mathcal{H}$. Using h_{t+1} we induce a sampling distribution over the pool $\mathcal{P}$, at the start of round $t+1$ (see step 4 of algorithm 1). As discussed in section 2.2, the probability distribution is such that it minimizes the (conditional) variance of the estimate of risk of h_t. We shall make this clear in section 3.2.

3 Risk Estimates and Confidence Bounds

We begin with the notation that will be required to develop our confidence bounds. Let p_i^t be the probability of querying x_i in round t, and $Q_i^t \in \{0,1\}$ be the random variable which takes the value 1 if x_i was queried in round t, and 0 otherwise. Hence $\mathbb{E}[Q_i^t|p_i^t] = 1$. For convenience, we shall denote by $Q_{1:n}^{1:t}$ the collection of random variables $Q_1^1, \ldots Q_1^t, \ldots, Q_n^1, \ldots, Q_n^t$. Let $Z_i^t \overset{\text{def}}{=} y_i Q_i^t$. Denote by $x_{1:n}$ the collection of random variables $x_1, \ldots, x_n$. Also let $[x]_+ = \max\{x, 0\}$.

We shall make the following independence assumption:

Assumption 1. *If x_i has not been queried up until the start of round t, then $p_i^t \perp\!\!\!\perp y_i | x_{1:n}, Z_{1:n}^{1:t-1}$.*

For any hypothesis $h \in \mathcal{H}$, define $\hat{L}_t(h) \overset{\text{def}}{=} \frac{1}{nt}\sum_{i=1}^n \sum_{\tau=1}^t \frac{Q_i^\tau}{p_i^\tau} L(y_i h(x_i))$. $\hat{L}_t(h)$ is an unbiased estimator of the risk of the hypothesis h, and was first proposed by Ganti and Gray (2011).

[4] An algorithm similar to the one suggested in this paper, can be designed such that re-querying is not allowed. This requires different type of estimators, and the expressions for $\mathrm{LCB}_t(h)$ turned out to be rather complicated. Hence, for simplicity of exposition we allow re-querying of points in this paper.

MAB	AL
Arms	Hypothesis
Loss signal on pulling an arm helps improve cumulative loss estimates	Sampling distribution designed to reduce variance of risk estimates of hypothesis

Table 1: The analogy between MAB and AL that is used as a guiding principle for the design of LCB-AL

Algorithm 1 LCB-AL Input: $\mathcal{P} = \{x_1, \ldots, x_n\}$, Loss function $L(\cdot)$, Budget B, Labeling Oracle $\mathcal{O}$, $p_{\min}$

```
1: Set $h_1 = 0, t = 1$.
2: while num_queried $\leq B$ do
3:   for $x_i \in \mathcal{P}$ do
4:     $$\bar{y}_i = \begin{cases} y_i & \text{if } x_i \in \mathcal{Q}_{t-1} \\ sign(h_t(x_i)) & \text{otherwise} \end{cases} \quad (1)$$
5:     $p_i^t \leftarrow p_{\min} + (1 - np_{\min}) \frac{L(\bar{y}_i h_t(x_i))}{\sum_{x_i \in \mathcal{P}} L(\bar{y}_i h_t(x_i))}$.
6:   end for
7:   Sample a point (say $x$) from the probability vector $p^t$.
8:   if $x$ was not queried in the past then
9:     Query $\mathcal{O}$ for the label $y$ of $x$.
10:    num_queried $\leftarrow$ num_queried + 1
11:  else
12:    Reuse the label of $x$.
13:  end if
14:  Solve: $h_{t+1} = \arg\min_{h \in \mathcal{H}} \text{LCB}_t(h) + \lambda_t \mathcal{R}(h)$.
15:  $t \leftarrow t + 1$
16: end while
17: Return $h_B$.
```

Finally let $\mathcal{Q}_t \overset{\text{def}}{=} \{x_i \in \mathcal{P} | \sum_{\tau=1}^t Q_i^\tau > 0\}$. Let $\mathcal{F}_\tau \overset{\text{def}}{=} \sigma(x_{1:n}, Z_{1:n}^{1:\tau})$ be the smallest sigma algebra that makes the random variables $x_{1:n}, Z_{1:n}^{1:\tau}$ measurable. Clearly $\mathcal{F}_1 \subset \ldots \subset \mathcal{F}_t$ form a filtration. Also we shall assume that our loss function is a convex function of the margin $yh(x)$, and is upper bounded by $L_{\max} < \infty$ for all $x \in \mathcal{P}, h \in \mathcal{H}$. Popular loss functions such as logistic loss, exponential loss, squared loss all satisfy these criteria.

3.1 Constructing lower confidence bounds

Utilizing the unbiased estimator $\hat{L}_t(h)$, along with Bernstein type inequalities for martingales allows us to construct lower confidence bounds for $R(h)$. We shall begin with the standard Azuma-Hoeffding bound for martingale difference sequences.

Theorem 1. *[Azuma-Hoeffding inequality] Let $X_1, X_2, \ldots$ be a martingale difference sequence w.r.t a filtration $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots$. If for each $i \geq 1$, $|X_i| \leq c_i$. Then,*

$$\mathbb{P}[|\sum_{i=1}^n X_i| \geq \epsilon] \leq 2 \exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^n c_i^2}\right)$$

We shall also need the following Bernstein type result from Bartlett et al. (2008).

Theorem 2. *Let $M_1, \ldots, M_t$ be a martingale difference sequence (MDS), w.r.t. the filtration $\mathcal{F}_1 \subset \ldots \subset \mathcal{F}_t$, with $|M_\tau| \leq b$. Let $\mathbb{V}_\tau M_\tau \overset{\text{def}}{=} \mathbb{V}(M_\tau|\mathcal{F}_{\tau-1})$, and $\sigma^2 \overset{\text{def}}{=} \sum_{\tau=1}^t \mathbb{V}_\tau M_\tau$. Then we have, for any $\delta < 1/e$, and $t \geq 4$, with probability at least $1 - \delta \log(t)$*

$$\sum_{\tau=1}^t M_\tau < 2 \max\{2\sigma, b\sqrt{\log(1/\delta)}\}\sqrt{log(1/\delta)}.$$

Lemma 1. *For any fixed $h \in \mathcal{H}$, $t \geq 4, \delta < 1/e$, with probability at least $1 - \delta \log(t)$, we have*

$$\frac{1}{n}\sum_{i=1}^n \sum_{\tau=1}^t \frac{Q_i^\tau}{p_i^\tau} L(y_i h(x_i)) - \frac{t}{n}\sum_{i=1}^n L(y_i h(x_i)) \leq$$
$$2\max\left(\frac{2}{n}\sqrt{\sum_{\tau=1}^t \sum_{i=1}^n \frac{L^2(y_i h(x_i))}{p_i^\tau} - \left(\sum_{i=1}^n L(y_i h(x_i))\right)^2},\right.$$
$$\left. L_{\max}\left(1 + \frac{1}{np_{\min}}\right)\sqrt{\log(1/\delta)}\right)\sqrt{\log(1/\delta)}$$

Proof. Let,

$$M_\tau \overset{\text{def}}{=} \frac{1}{n}\sum_{i=1}^n \frac{Q_i^\tau}{p_i^\tau} L(y_i h(x_i)) - \frac{1}{n}\sum_{i=1}^n L(y_i h(x_i)). \quad (2)$$

Utilizing the independence assumption it is easy to see that $\mathbb{E}[M_\tau|\mathcal{F}_{\tau-1}] = 0$. Hence $M_1, \ldots, M_t$ form a martingale difference sequence w.r.t. the filtration $\mathcal{F}_1, \ldots, \mathcal{F}_t$. In order to apply theorem 2 we need estimates for the sum of conditional variances, and the range of $|M_\tau|$. We proceed to establish upper bounds on these quantities now.

From equation 2 and triangle inequality we get

$$|M_\tau| \leq \frac{1}{n}|\sum_{i=1}^n \frac{Q_i^\tau}{p_i^\tau} L(y_i h(x_i))| + \frac{1}{n}|\sum_{i=1}^n L(y_i h(x_i))|$$
$$\leq L_{\max}\left(1 + \frac{1}{np_{\min}}\right). \quad (3)$$

$$\sigma^2 \overset{\text{def}}{=} \sum_{\tau=1}^{t} \mathbb{E}[M_\tau^2|\mathcal{F}_{\tau-1}]$$
$$= \sum_{\tau=1}^{t} \frac{1}{n^2}\mathbb{E}\Big[\frac{Q_i^\tau}{(p_i^\tau)^2}L^2(y_ih(x_i)) + \underbrace{\frac{2}{n}\sum_{i\neq j}\frac{Q_i^\tau Q_j^\tau}{p_i^\tau p_j^\tau}L(y_ih(x_i))L(y_jh(x_j))}_{=0} \tag{4}$$
$$- \frac{1}{n^2}\Big(\sum_{i=1}^{n} L(y_ih(x_i))\Big)^2|\mathcal{F}_{\tau-1}\Big]$$
$$= \frac{1}{n^2}\sum_{\tau=1}^{t}\sum_{i=1}^{n}\frac{L^2(y_ih(x_i))}{p_i^\tau} - \frac{1}{n^2}\Big(\sum_{i=1}^{n} L(y_ih(x_i))\Big)^2. \tag{5}$$

In equation 4 we used the fact that in each round only one point is queried. The result now follows by the application of theorem 2 with b, σ^2 defined as in equations 3, 5 respectively. □

While this theorem enables us to construct lower bounds for the risk of the hypothesis, the major problem is that the RHS of theorem 1 depends on the labels of all the points in the pool and not just the queried labels. We shall now provide an estimator for the variance term σ^2.

Lemma 2. *With probability at least* $1-\delta$, *we have*

$$\sigma^2 \leq \frac{1}{n^2}\Big[\sum_{\substack{i=1:n\\ \tau=1:t}}\frac{Q_i^\tau}{(p_i^\tau)^2}L^2(y_ih(x_i)) - \Big(\sum_{\mathcal{Q}_t} L(y_ih(x_i))\Big)^2 + \frac{L_{\max}^2\sqrt{2t\log(1/\delta)(n-1)}}{\sqrt{p_{\min}}}\Big]_+$$

Proof. From the proof of theorem 1 we have

$$\sigma^2 = \underbrace{\frac{1}{n^2}\sum_{\tau=1}^{t}\sum_{i=1}^{n}\frac{L^2(y_ih(x_i))}{p_i^\tau}}_{I_1} - \underbrace{\frac{1}{n^2}\Big(\sum_{i=1}^{n} L(y_ih(x_i))\Big)^2}_{I_2}.$$

A simple lower bound on I_2 is

$$\hat{I}_2 \overset{\text{def}}{=} \frac{1}{n^2}\Big(\sum_{\mathcal{Q}_t} L(y_ih(x_i))\Big)^2. \tag{6}$$

Now let

$$\hat{I}_1 \overset{\text{def}}{=} \frac{1}{n^2}\sum_{\tau=1}^{t}\sum_{i=1}^{n}\frac{Q_i^\tau}{(p_i^\tau)^2}L^2(y_ih(x_i)).$$

Define

$$M_\tau \overset{\text{def}}{=} \frac{1}{n^2}\sum_{i=1}^{n}\frac{Q_i^\tau}{(p_i^\tau)^2}L^2(y_ih(x_i)) - \frac{1}{n^2}\sum_{i=1}^{n}\frac{1}{p_i^\tau}L^2(y_ih(x_i))$$

Once again utilizing our independence assumption, we conclude that $M_1, \dots M_t$ form an MDS w.r.t. the filtration $\mathcal{F}_1, \dots, \mathcal{F}_t$. Applying theorem 1 to this MDS, we get with probability at least $1-\delta$

$$|\sum_{\tau=1}^{t} M_\tau| \leq \frac{L_{\max}^2\sqrt{2t\log(1/\delta)}}{n^2}\sqrt{\frac{n-1}{p_{\min}}}. \tag{7}$$

The result follows from equations 6, 7 □

We are now ready to establish a lower confidence bound on the risk of hypotheses in $\mathcal{H}$.

Theorem 3. *Let* $|\mathcal{H}| < \infty$. *With probability at least* $1 - |\mathcal{H}|\delta(2 + T\log(T/e))$, *for all* $h \in \mathcal{H}, 4 \leq t \leq T$, *and* $\delta < 1/e$, *we have*

$$R(h) \geq \Big[\hat{L}_t(h) - \frac{2}{t}\log(1/\delta)L_{\max}\Big(1 + \frac{1}{np_{\min}}\Big) - \frac{4}{nt}\sqrt{V_t\log(1/\delta)} - \sqrt{\frac{L_{\max}^2\log(1/\delta)}{2n}}\Big]_+ \tag{8}$$

where

$$V_t \overset{\text{def}}{=} \Big[\sum_{\substack{i=1:n\\ \tau=1:t}}\frac{Q_i^\tau}{(p_i^\tau)^2}L^2(y_ih(x_i)) - \Big(\sum_{\mathcal{Q}_t} L(y_ih(x_i))\Big)^2 + \frac{L_{\max}^2\sqrt{2t\log(1/\delta)(n-1)}}{\sqrt{p_{\min}}}\Big]_+ \tag{9}$$

Proof. For any fixed $h \in \mathcal{H}$, and $t \leq T$, we have from theorems 1, 2, Hoeffding inequality, and the union bound that with probability at least $1 - \delta\log(t) - 2\delta$

$$R(h) \geq \Big[\hat{L}_t(h) - \frac{2}{t}\log(1/\delta)L_{\max}(1 + \frac{1}{np_{\min}}) - \frac{4}{nt}\sqrt{V_t\log(1/\delta)} - \sqrt{\frac{L_{\max}^2\log(1/\delta)}{2n}}\Big]_+. \tag{10}$$

Applying union bound over all hypothesis and over all $t \geq 4$, and approximating $n-1$ with n we get the desired result. □

Specification of $\text{LCB}_t(h)$. Theorem 3 provides us with an expression for $\text{LCB}_t(h)$. For the purpose of solving the optimization in step 7 of our LCB-AL algorithm, we can set

$$\text{LCB}_t(h) \overset{\text{def}}{=} \hat{L}_t(h) - \frac{4}{nt}\sqrt{\log(1/\delta)V_t}, \tag{11}$$

where V_t is shown in equation 9.

3.2 Query probability distribution in each round of LCB-AL

The only thing that is left to be motivated in LCB-AL is the choice of probability distribution in step 3. As explained in section 2.2 we want to use a sampling distribution, such that the conditional variance of the risk estimate $\hat{L}_t(h)$ of h_t is minimized. We shall now show how the sampling distribution should be designed in order to achieve this goal. Let $\Delta \subset \mathbb{R}^n_+$ be the probability simplex. Let $V_t(\cdot)$ denote the variance, conditioned on $x_{1:n}, Z^{1:t-1}_{1:n}$. Let $p^t \overset{\text{def}}{=} (p^t_1, \ldots, p^t_n) \in \Delta$. At the start of round t, the desired sampling distribution, p^t should satisfy

$$p^t = \arg\min_{\tilde{p}^t \in \Delta} \mathbb{V}_t \underbrace{\Big[\frac{1}{nt} \sum_{\substack{i=1:n \\ \tau=1:t}} \frac{\tilde{Q}^\tau_i}{\tilde{p}^\tau_i} L(y_i h_t(x_i))\Big]}_{\hat{L}_t(h_t)}$$

$$= \arg\min_{\tilde{p}^t \in \Delta} \sum_{i=1}^{n} \frac{\mathbb{E}_t L^2(y_i h_t(x_i))}{\tilde{p}^\tau_i}$$

Solving the above optimization problem yields the simple solution $p^t_i \propto \sqrt{\mathbb{E}_t L^2(y_i h_t(x_i))}$. If $x_i \in \mathcal{Q}_{t-1}$, then the label y_i is known and hence, we let $p^t_i \propto L(y_i h_t(x_i))$. If $x_i \notin \mathcal{Q}_{t-1}$, then since y_i is yet unknown, we let $p^t_i \propto L(|h_t(x_i)|)$. This is equivalent to taking y_i to be equal to $\text{sgn}(h_t(x_i))$ (see steps 3, 4 of algorithm 1). This scheme encourages querying points which have small margin w.r.t the current classifier, h_t, or points which have already been queried for their label, but on which the current hypothesis, h_t suffers a large loss. In any round, the minimum probability of querying any point is $p_{\min}$. This guarantees that $\hat{L}_t(h)$ is an unbiased estimator of risk of h.

4 Related Work

A variety of pool based AL algorithms have been proposed in the literature employing various query strategies. Some of the popular querying strategies include uncertainty sampling, where the active learner queries the point whose label it is most uncertain about (Lewis and Gale, 1994; Settles and Craven, 2008; Tong and Chang, 2001). Usually the uncertainty in the label is calculated using certain information-theoretic criteria such as entropy, or variance of the label distribution. Seung et al. (1992) introduced the query-by-committee (QBC) framework where a committee of potential models, which all agree on the currently labeled data is maintained and, the point where most committee members disagree is considered for querying. Other frameworks include querying the point, which causes the maximum expected reduction in error (Zhu et al., 2003; Guo and Greiner, 2007), variance reducing query strategies such as the ones based on optimal design (Flaherty et al., 2005; Zhang and Oles, 2000). A very thorough literature survey of different active learning algorithms has been done by Settles (2009). AL algorithms that are consistent and have provable label complexity have been proposed for the agnostic setting for the 0-1 loss in recent years (Dasgupta et al., 2007; Balcan et al., 2009). Hanneke and Yang (2012) recently provided a disagreement region based algorithm, with provable guarantees for active learning with general loss functions.

Algorithmically, LCB-AL is similar in flavor to the UPAL algorithm introduced by Ganti and Gray (2011). In the UPAL algorithm the authors suggested minimizing an unbiased estimator of risk of h, and a sampling distribution that was in proportion to the entropy of the prediction on the pool. However as we suggested the use of self-concordant regularizer is very crucial in tackling the high variance of our estimators. As we show in our experiments (see section 5) the use of self-concordant barrier as a regularizer, helps lend stability to our algorithm, and consequently LCB-AL performs better than UPAL.

To our knowledge there has been only one other paper bridging the world of active learning and MAB. Baram et al. (2004) proposed a meta-active learning algorithm called COMB. COMB was an implementation of the EXP4 algorithm for MAB with expert advice, where the different active learning algorithms are the various "experts" and the different points in the pool are the arms of the MAB. Briefly, in each round, each of the experts suggest a sampling distribution on the pool. COMB maintains an estimate of the error rate of each expert, and uses exponential weighting to come up with a sampling distribution on the pool. In order to estimate the error rate of each of the experts, the authors proposed a proxy reward function of querying a point in terms of the entropy of label distribution of the unlabeled pool, induced by the classifier obtained on the labeled dataset gathered by COMB till the current iteration. In a way, the concept of reward seems inevitable in their formulation because the unlabeled points in the pool are treated as arms of the MAB. In contrast, we think of the arms of the bandit as the different hypothesis, and querying a data point, as the process of improving our estimate of the risk of the different hypothesis. Hence, we bypass the need for an explicit reward signal, yet utilize MAB ideas for AL.

5 Experiments

We implemented LCB-AL in MATLAB, and compared it with some previously proposed active learning algorithms on four UCI datasets. The competing algorithms are UPAL (Ganti and Gray, 2011), BMAL (Hoi et al., 2006), and a passive learning (PL) algorithm that minimizes the regularized logistic loss. UPAL is a sequential, pool based algorithm that minimizes the unbiased estimator $\hat{L}_t(h)$, of the risk of a hypothesis h, along with the squared L_2 norm of h in each round. BMAL is a batch mode active learning algorithm introduced by Hoi et al. (2006). Hoi et al. in their paper showed superior empirical performance of BMAL over other pool based active learning algorithms, and this is the primary motivation for using BMAL in our experiments. Given a pool $\mathcal{P}$, and a budget B, BMAL chooses a set of B points that minimize the Fisher information ratio between the set of unqueried and the queried points. The authors then propose a monotonic submodular approximation to the original Fisher ratio objective, which is optimized by a greedy algorithm. In order to keep our experiments simple, and to facilitate easy comparison, we restricted our hypothesis class to the set of linear hypothesis of bounded L_2 norm $\mathcal{H} = \{h : ||h|| \leq R\}$. For this set $\mathcal{H}$, the self-concordant regularizer $\mathcal{R}(h)$ is equal to $-\log(R^2 - ||h||^2)$ (Abernethy et al., 2008), where $R > 0$ was provided as an input to LCB-AL. For our implementation of LCB-AL, we used a slightly different definition for $\text{LCB}_t(h)$, than the one proposed in equation 11. Let

$$\text{LCB}'_t(h) \overset{\text{def}}{=} \hat{L}_t(h) - C_t\sqrt{V'_t} - \lambda_t \log(R^2 - ||h||^2), \quad (12)$$

where

$$V'_t \overset{\text{def}}{=} \Big[\sum_{\substack{i=1:n \\ \tau=1:t}} \frac{Q_i^\tau}{(p_i^\tau)^2} L^2(y_i h(x_i)) - \Big(\sum_{\mathcal{Q}_t} L(y_i h(x_i))\Big)^2\Big]_+ .$$

The definition of $\text{LCB}'_t(h)$ is almost similar to the one suggested by equation 11 except that the terms that were independent of h, in $\text{LCB}_t(h)$ were dropped to get $\text{LCB}'_t(h)$. C_t, and λ_t in all our experiments were set to $\frac{\sqrt{\log(t)}}{10}$, and $\frac{100nt}{\left(\sum_{i=1}^n \sum_{\tau=1}^{t-1} \frac{Q_i^\tau}{p_i^\tau}\right)^{1/3}}$ respectively. We used minFunc [5] to solve all of our optimization problems. We used a budget of 300 points. Finally, each of the dataset was scaled to $[-1, 1]^d$. Since, UPAL and LCB-AL are randomized algorithms, on each dataset, we ran them 10 times each, and report averaged measurements.

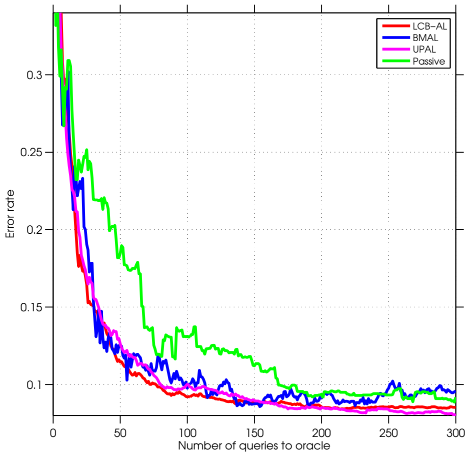
5.1 Experimental comparisons of different algorithms

Figure 1 shows the test error of the hypothesis obtained, corresponding to the number of unique queries made to the oracle, by each algorithm. Table 2 shows the error rate on the test set, of each algorithm, once the budget is exhausted. On three of the datasets, namely MNIST, Abalone, and Statlog utilizing an active learner is better than a passive learner. On MNIST, the performance of LCB-AL and UPAL are nearly equal as far as the final test error goes, and both are better than BMAL. On abalone LCB-AL is better than both BMAL and UPAL, while on Statlog the final error achieved by BMAL is better than UPAL, and also LCB-AL, though the difference between LCB-AL and BMAL is pretty narrow. On Whitewine, passive learner is better than any of the active learners. In order to gain an insight into how well each of the learning algorithm learns with each query to the oracle, we also report the cumulative error rate of each algorithm, summed over all the queries. The cumulative error rate of a learning algorithm is nothing but the area under the curve (AUC) of error-rate vs number of queries to the oracle. Even on this measure, LCB-AL and UPAL are better than BMAL on MNIST, Abalone and Whitewine datasets. On Abalone, and Statlog the AUC of LCB-AL is appreciably smaller than that of UPAL.

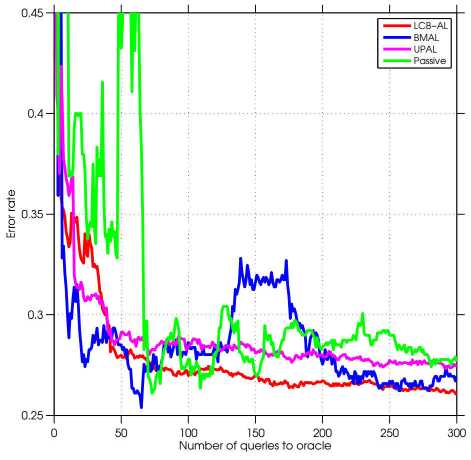
5.2 Comparing UPAL with LCB-AL

From our first set of experiments it looks like UPAL is just as good as LCB-AL if not any better. e.g. on the MNIST dataset, there is almost no difference between LCB-AL and UPAL. Since, both LCB-AL and UPAL are randomized algorithms it makes sense to measure the fluctuations in the performance of both the algorithms. Table 3 gives the standard deviation of AUC over all the runs for both LCB-AL and UPAL. It is clear that the standard deviation of AUC for LCB-AL is uniformly smaller than that of UPAL over all datasets, and the difference in the standard deviations is largest for the MNIST dataset. This can be explained by the fact that, the unbiased estimator of risk used in UPAL is a high variance estimator, and hence not a reliable estimator of the risk of a hypothesis. In LCB-AL, by utilizing lower confidence bounds, and the self-concordant regularizer, we are able to tackle the high variance of our estimator, and at the same time harness the variance for exploration in the hypothesis space. In fact, a similar phenomenon occurs even in the MAB setting, where algorithms built only on unbiased estimators, such as EXP3 (Auer et al., 2002b), achieve optimal performance only on an average, whereas algorithm using confidence bounds such as EXP3.P achieve optimal performance with high probability.

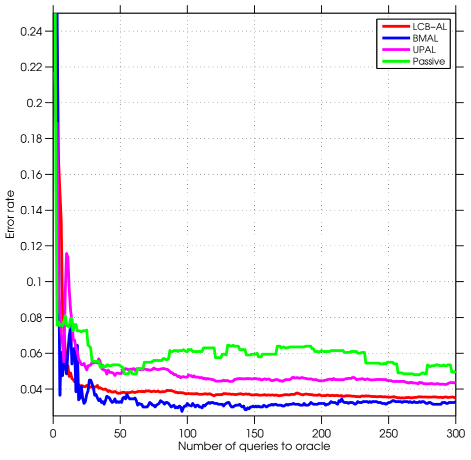
[5] minFunc can be downloaded from `http://www.di.ens.fr/~mschmidt/Software/minFunc.html`



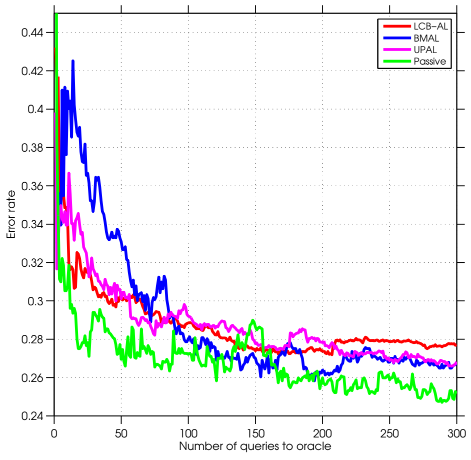
(a) MNIST (3 vs 5)



(b) Abalone



(c) Statlog



(d) Whitewine

Figure 1: Error rate of different learning algorithms with the number of queries made to the oracle.

Dataset	LCB-AL		UPAL		BMAL		PL	
MNIST	0.0808	33.27	0.0809	32.75	0.0958	34.89	0.0918	40.08
Abalone	0.2604	83.49	0.2747	86.60	0.2695	86.21	0.2766	93.60
Statlog	0.0354	12.59	0.0433	14.97	0.0330	11.33	0.05	18.06
Whitewine	0.2771	86.30	0.2682	86.21	0.2665	86.95	0.2517	80.94

Table 2: Comparison of various active learning algorithms and passive learner on various datasets.In this table we report both the error rate of each learner after it has exhausted its budget, as well as the area under the curve of error rate vs number of queries made for each learning algorithm.

Dataset	LCB-AL	UPAL
MNIST	3.8604	5.0132
Abalone	2.6512	2.6869
Statlog	0.7944	1.6691
Whitewine	2.4097	2.9992

Table 3: Standard deviation of the AUC of LCB-AL and UPAL on different datasets.

6 Discussion

We proposed LCB-AL a multi-armed bandit inspired pool based active learning algorithm. By viewing the problem of active learning as quickly detecting the hypothesis with (near) optimal risk, we view the problem of active learning as similar to a MAB problem with the arms being the different hypothesis. By building lower confidence bounds on the risk of each hypothesis we are able to perform exploration in the hypothesis space. By conceptually investigating the role of a loss signal in MAB, we are able to design a sampling distribution from which we sample the points to be queried. Experimental results suggest that our algorithm is both more accurate, and also more stabler than competing active learning algorithms.

In the near future we would like to investigate LCB-AL theoretically. Two properties of LCB-AL are worth investigating. Firstly, can we guarantee that the excess risk of our algorithm goes to 0, as $n \to \infty, B \to \infty$? Secondly, what is the budget B, required in order to guarantee an excess risk of ϵ?

An immediate extension of this work could be to investigate how different concentration inequalities can be utilized to give different lower confidence bounds for the risk of a hypothesis. This has proven to be an attractive idea in the MAB setting and it is generally accepted that tighter concentration inequalities lead to better algorithms for MAB (Audibert et al., 2009a; Salomon and Audibert, 2011). We would expect something similar to happen even in AL.

On a more high level we believe that there is tremendous potential for ideas from multi-armed bandits, and various other extensions of multi-armed bandits such as contextual bandits, bandit optimization, to be used for active learning problems. Most of these algorithms are very simple, efficient and hence should be useful in designing simple, efficient active learning algorithms.

References

J. Abernethy, E. Hazan, and A. Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, volume 3, page 3, 2008.

Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. *Theor. Comput. Sci.*, 2009a.

J.Y. Audibert, R. Munos, and C. Szepesvári. Exploration–exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410(19):1876–1902, 2009b.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002a.

P. Auer, N. Cesa-Bianchi, Y. Freund, and R.E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b.

M.F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. *JCSS*, 75(1), 2009.

Y. Baram, R. El-Yaniv, and K. Luz. Online choice of active learning algorithms. *The Journal of Machine Learning Research*, 5:255–291, 2004.

P.L. Bartlett, V. Dani, T. Hayes, S. Kakade, A. Rakhlin, and A. Tewari. High-probability regret bounds for bandit online linear optimization. *COLT*, 2008.

E.B. Baum and K. Lang. Query learning can work poorly when a human oracle is used. In *IJCNN*, 1992.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *arXiv preprint arXiv:1204.5721*, 2012.

W. Chu, M. Zinkevich, L. Li, A. Thomas, and B. Tseng. Unbiased online active learning in data streams. In *SIGKDD*, 2011.

D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15 (2), 1994.

S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. *NIPS*, 2007.

Patrick Flaherty, Michael I. Jordan, and Adam P. Arkin. Robust design of biological experiments. In *Neural Information Processing Systems*, 2005.

R. Ganti and A. Gray. Upal: Unbiased pool based active learning. *Arxiv preprint arXiv:1111.1784*, 2011.

Y. Guo and R. Greiner. Optimistic active learning using mutual information. In *IJCAI*, 2007.

S. Hanneke and L. Yang. Surrogate losses in passive and active learning. *arXiv preprint arXiv:1207.3772*, 2012.

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*, 2006.

D.D. Lewis and W.A. Gale. A sequential algorithm for training text classifiers. In *SIGIR*, 1994.

Yurii Nesterov and A Nemirovsky. Interior point polynomial methods in convex programming, 1994.

Antoine Salomon and Jean-Yves Audibert. Deviations of stochastic bandit regret. In *Algorithmic Learning Theory*, pages 159–173. Springer, 2011.

B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *EMNLP*, 2008.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

H.S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *COLT*, pages 287–294. ACM, 1992.

S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia*, 2001.

T. Zhang and F. Oles. The value of unlabeled data for classification problems. In *ICML*, 2000.

Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003.

Batch-iFDD for Representation Expansion in Large MDPs

Alborz Geramifard[†] **Thomas J. Walsh**[†] **Nicholas Roy**[⋆] **Jonathan P. How**[†]

[†]Laboratory for Information and Decision Systems
[⋆]Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
77 Massachusetts Ave., Cambridge, MA 02139

Abstract

Matching pursuit (MP) methods are a promising class of feature construction algorithms for value function approximation. Yet existing MP methods require creating a pool of potential features, mandating expert knowledge or enumeration of a large feature pool, both of which hinder scalability. This paper introduces batch incremental feature dependency discovery (Batch-iFDD) as an MP method that inherits a provable convergence property. Additionally, Batch-iFDD does not require a large pool of features, leading to lower computational complexity. Empirical policy evaluation results across three domains with up to one million states highlight the scalability of Batch-iFDD over the previous state of the art MP algorithm.

1 Introduction

In complex decision-making tasks, from stacking blocks to flying surveillance missions, the number of possible features used to represent a domain grows exponentially in the basic number of variables. It follows that generating a small number of relevant features that are sufficient for determining an optimal policy is a critical component for tractable reinforcement learning in complex environments. However, even in the well-studied case of linear value function approximation [Silver *et al.*, 2008; Stone *et al.*, 2005], finding the "right" set of features remains a challenge.

In the linear value function approximation case, several methods exist for automated feature construction [*e.g.,* Lin and Wright, 2010; Ratitch and Precup, 2004; Whiteson *et al.*, 2007]. Of these techniques, Matching Pursuit (MP) algorithms [Painter-Wakefield and Parr, 2012] have shown great promise in incrementally expanding the set of features to better model the value function. However, all prior MP techniques begin with a collection of potential features from which new features are selected. In large MDPs, this pool of features has to either be carefully selected by a domain expert or be prohibitively large to include all critical features, both hindering scalability. Still, despite such requirements, MP algorithms have various desirable properties [See Painter-Wakefield and Parr, 2012], making them an attractive option for RL feature construction.

Some similar techniques avoid enumerating a set of potential features, but are not scalable for other reasons. For example, Bellman Error Basis Function (BEBF) [Parr *et al.*, 2007] iteratively constructs a set of basis vectors without an enumerated pool of potential features. However, BEBF relies on supervised learning techniques to map states to their feature values. This process can be as complex as determining the value function itself, mitigating the tractability gains of feature construction. Similarly, Proto-Value Functions [Mahadevan *et al.*, 2006] do not use a pool of potential features but learn a complex manifold representation that can be computationally intensive for arbitrary MDPs.

This paper presents a new algorithm, Batch incremental Feature Dependency Discovery (Batch-iFDD), that does not require a large set of potential features at initialization. Moreover, we prove Batch-iFDD is an MP algorithm, thereby inheriting the theoretical benefits associated with those techniques. Batch-iFDD extends the previously described online incremental Feature Dependency Discovery (iFDD) algorithm [Geramifard *et al.*, 2011], which creates increasingly finer features that help to eliminate error of the value function approximation. Our contributions in this paper are to (1) extend iFDD to the batch setting (Section 2.4), (2) prove that Batch-iFDD is an MP algorithm (Corollary 3.6) and derive its guaranteed rate of error-bound reduction (Theorem 3.4), (3) derive a practical approximation for iFDD's objective function resulting in an algorithm called Batch-iFDD$^+$ (Equation 10), and (4) empirically compare Batch-iFDD with the state of the art MP algorithm across three domains including a 20 dimensional version of the System Administrator domain with over one million states (Section 4).

2 Preliminaries

In this section we define data structures for modeling RL domains and approximating value functions. We also describe a basic reinforcement learning technique (Temporal Difference Learning) for evaluating a policy's value through experience. Finally, we provide definitions of the relationships between features and describe the feature search process.

2.1 Reinforcement Learning

A **Markov Decision Process (MDP)** is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}^a_{ss'}, \mathcal{R}^a_{ss'}, \gamma)$ where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, $\mathcal{P}^a_{ss'}$ is the probability of getting to state s' by taking action a in state s, $\mathcal{R}^a_{ss'}$ is the corresponding reward, and $\gamma \in [0, 1)$ is a discount factor that balances current and future rewards . We focus on MDPs with finite states. A *trajectory* is a sequence $s_0, a_0, r_0, s_1, a_1, r_1, s_2, \ldots$, where the action $a_t \in \mathcal{A}$ is chosen according to a deterministic *policy* $\pi : \mathcal{S} \to \mathcal{A}$, mapping each state to an action. Given a policy π, the value function, $V^\pi(s)$ for each state, is the expected sum of the discounted rewards for an agent starting at state s and then following policy π thereafter:

$$\begin{aligned} V^\pi(s) &= E_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| s_0 = s\right] \\ &= \sum_{s' \in \mathcal{S}} \mathcal{P}^{\pi(s)}_{ss'} \left[\mathcal{R}^{\pi(s)}_{ss'} + \gamma V^\pi(s')\right]. \end{aligned}$$

Since this paper primarily addresses the policy evaluation problem (*i.e.,* finding the value function of a fixed policy), the π notation is dropped from this point on and included implicitly. For a finite-state MDP, the vector $\boldsymbol{V}_{|\mathcal{S}|\times 1}$ represents the value function. The matrix $\boldsymbol{P}_{|\mathcal{S}|\times|\mathcal{S}|}$ represents the transition model with $\boldsymbol{P}_{ij} = \mathcal{P}^{\pi(s_i)}_{s_i s_j}$, and the vector $\boldsymbol{R}_{|\mathcal{S}|\times 1}$ is the reward model, with $\boldsymbol{R}_i = \sum_j \mathcal{P}^{\pi(s_i)}_{s_i s_j} \mathcal{R}^{\pi(s_i)}_{s_i s_j}$. Hence $\boldsymbol{V}$ can be calculated in the matrix form as $\boldsymbol{V} = \boldsymbol{R} + \gamma \boldsymbol{P}\boldsymbol{V} = \mathbf{T}(\boldsymbol{V})$, where $\mathbf{T}$ is the Bellman operator.

Storing a unique value for each state is impractical for large state spaces. A common approach is to use a linear approximation of the form $V(s) = \boldsymbol{\theta}^\top \phi(s)$. The feature function $\phi : \mathcal{S} \to \mathbb{R}^n$ maps each state to a vector of scalar values. Each element of the feature function $\phi(s)$ is called a *feature*; $\phi_f(s) = c \in \mathbb{R}$ denotes that feature f has scalar value c for state s, where $f \in \boldsymbol{\chi} = \{1, \ldots, n\}$. $\boldsymbol{\chi}$ represents the set of features;[1] the vector $\boldsymbol{\theta} \in \mathbb{R}^n$ holds weights. Hence,

$$\boldsymbol{V} \approx \tilde{\boldsymbol{V}} = \begin{bmatrix} — \phi^\top(s_1) — \\ — \phi^\top(s_2) — \\ \vdots \\ — \phi^\top(s_{|\mathcal{S}|}) — \end{bmatrix} \times \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \triangleq \boldsymbol{\Phi}\boldsymbol{\theta}.$$

Binary features ($\phi_f : \mathcal{S} \to \{0, 1\}$) are of special interest to practitioners [Silver *et al.*, 2008; Stone *et al.*, 2005; Sturtevant and White, 2006], mainly because they are computationally cheap, and are the focus of this paper.

The **Temporal Difference Learning (TD)** [Sutton, 1988] algorithm is a traditional policy evaluation method where the current $V(s)$ estimate is adjusted based on the difference between the current estimated state value and a better approximation formed by the actual observed reward and the estimated value of the following state. Given (s_t, r_t, s_{t+1}) and the current value estimates, the TD-error, δ_t, is calculated as: $\delta_t(V) = r_t + \gamma V(s_{t+1}) - V(s_t)$. The one-step TD algorithm, also known as TD(0), updates the value estimates using $V(s_t) = V(s_t) + \alpha\delta_t(V)$, where α is the learning rate. In the case of linear function approximation, the TD update can be used to change the weights of the value function approximator: $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\delta_t(V)$. In the batch setting, the least-squares TD (LSTD) algorithm [Bradtke and Barto, 1996] finds the weight vector directly by minimizing the sum of TD updates over all the observed data.

2.2 Matching Pursuit Algorithms

This paper considers algorithms that expand features during the learning process. The class of matching pursuit (MP) algorithms, such as OMP-TD [Painter-Wakefield and Parr, 2012] has been shown recently to be a promising approach for feature expansion. An algorithm is MP if it selects the new feature from the pool of features that has the highest correlation with the residual.

2.3 Finer (Coarser) Features and Search

We now define some properties of state features and feature-search algorithms. The *coverage* of a feature is the portion of the state space for which the feature value is active (*i.e.,* non-zero). We say that a feature A is *coarser* than feature B (B is *finer* than A) if A has a higher coverage than B. For example, consider a task of administrating 3 computers (C_1, C_2 and C_3) that each can be up or down. Feature A (C_1 = down) is coarser than feature B (C_3= down AND C_2 = up), because coverage(A) = 0.5 > coverage(B) = 0.25.

[1]Properties such as being close to a wall or having low fuel can be considered as features. In our setting, we assume all such properties are labeled with numbers and are addressed with their corresponding number.

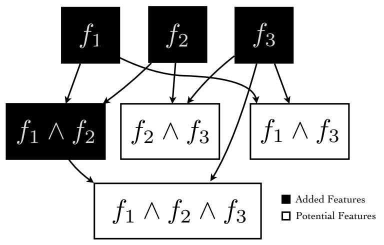


Figure 1: A partial concept lattice showing potential features as identified by iFDD. With methods like OMP-TD, all nodes are considered as potential features, despite their location in the lattice.

High coverage is not always the best criterion for selecting features because the resulting partitions may not yield a good approximation of the value function. For example, suppose that having C_1 = down translates into negative values except when C_3 = up. If the weight corresponding to feature A (C_1 = down) is set to a negative value, the value function is reduced for all parts of the state space covered by A, including situations where C_3 = up. The problem of feature construction is to find a small set of features with high coverage that still approximate the value function with bounded error.

One approach is to assume a set of *base* binary features that, in full combination, uniquely describe each state. These features would have high coverage, and constitute a very large set. Combining base features with the conjunction operator would allow us to find features with lower coverage. This search process can be thought of as selecting nodes in a graph structure (Figure 1). Each node represents a feature, with the top level nodes corresponding to the domain's base features. An edge indicates the origin feature subsumes the destination feature (and therefore the destination is finer). For instance, in our example, $((C_1 = \text{up}) \rightarrow (C_1 = \text{up} \wedge C_2 = \text{up}))$ would be an edge.

Current MP methods such as OMP-TD require an enumerated set of potential features (*e.g.,* all possible feature conjunctions of base features) that can be combinatorially large. MP methods iterate over this set on every step of feature generation, leading to high computational demand if the set is large, or poor performance if only a few nodes in the lattice are considered. Methods such as BEBF [Parr *et al.*, 2007] adopt an alternative approach by creating new features using supervised learning that often do not have a clean logical interpretation and can be arbitrary complex. Hence, this paper focuses on mapping states to conjunctive features in the concept lattice. Ideally, a search method would find relevant features in the lattice by selectively growing the tree as opposed to current MP techniques that have to be initialized with the set of all potential features.

2.4 iFDD and Batch-iFDD

The iFDD algorithm [Geramifard *et al.*, 2011] is an online feature expansion technique with low computational complexity. Given an initial set of base features, iFDD adds the conjunction of existing binary features as new features. Following the original work, we restrict new features to be the conjunction of previously selected features, which still gives us a rich set of potential features without introducing complex reasoning into the search process. At each time-step, iFDD performs the following steps:

1. Identify pair-wise combinations of active features.
2. Accumulate the absolute value of the observed TD-error for all such pairs.
3. If the accumulated value for a pair of features f and g exceeds a predefined threshold, add feature $f \wedge g$ to the pool of features.

Within the concept lattice described in Section 2.3, the first step considers features where two parent concepts are already in the feature space, and the conjunction of these parents is equivalent to the potential feature. Then potential features that also reduce the value function approximation error significantly are added to the feature set. This algorithm has the ability to include fine-grained features where they are necessary to represent the value function, but can avoid other (less helpful) features of similar complexity.

This paper uses iFDD for policy evaluation in a batch setting where LSTD estimates the TD-error over all samples and then the most "relevant" feature is added to the features. We now analyze the behavior of Batch-iFDD and through this analysis derive a new algorithm, Batch-iFDD$^+$, which better approximates the best guaranteed rate of error-bound reduction.

3 Theoretical Results

Geramifard *et al.* [2011] introduced iFDD, empirically verified the approach, and proved the asymptotic convergence of iFDD combined with TD. This work extends those theoretical results by showing that executing iFDD combined with TD in the batch setting is equivalent to approximately finding the feature from the conjunction of existing features with the maximum guaranteed error-bound reduction.

In order to use the conjunction operator to define new features, we redefine the notion of a feature. Given an initial feature function ϕ outputting vectors in $\mathbb{R}^n$, we address each of its n output elements as an *index* rather than a *feature* from this point on; $\phi_i(s) = c$ denotes index i of the initial feature function has value c in state s.[2] The set of all indices is $\mathcal{V}_n = \{1, \cdots, n\}$. A feature, f, is redefined as an arbitrary subset of $\mathcal{V}_n$, where $\phi_f(s) \triangleq \bigwedge_{i \in f} \phi_i(s)$, and

$$\phi_{\{\}}(s) \triangleq \begin{cases} 1 & \text{if } \forall i \in \mathcal{V}_n, \phi_i(s) = 0 \\ 0 & \text{otherwise} \end{cases},$$

where $\{\}$ is the null feature. For example $\phi_{\{1,3\}}(.) = \phi_1(.) \wedge \phi_3(.)$. Notice that a single index can constitute a feature (*e.g.,* $f = \{1\}$). Further we assume that all sets are ordered based on the cardinality size of each element in ascending order, unless specified. Given a set of features $\boldsymbol{\chi} = \{f_1, f_2, ...f_N\}$, the definition of $\boldsymbol{\Phi}$ is extended as follows:

$$\boldsymbol{\Phi}_{\boldsymbol{\chi}} = \begin{bmatrix} \phi_{f_1}(s_1) & \phi_{f_2}(s_1) & \cdots & \phi_{f_N}(s_1) \\ \phi_{f_1}(s_2) & \phi_{f_2}(s_2) & \cdots & \phi_{f_N}(s_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{f_1}(s_{|\mathcal{S}|}) & \phi_{f_2}(s_{|\mathcal{S}|}) & \cdots & \phi_{f_N}(s_{|\mathcal{S}|}) \end{bmatrix}_{|\mathcal{S}| \times N}.$$

For brevity, we define $\boldsymbol{\phi}_f$ as a column of feature matrix $\boldsymbol{\Phi}_{\boldsymbol{\chi}}$ that corresponds to feature f. $\boldsymbol{\phi}_f = \boldsymbol{\Phi}_{\{f\}} = \begin{bmatrix} \phi_f(s_1) & \phi_f(s_2) & \cdots & \phi_f(s_m) \end{bmatrix}^\top$. Also, define $B_n \triangleq \{\{\}, \{1\}, \{2\}, \cdots, \{n\}\}$, as the set of features with cardinality less than 2, and $\mathcal{F}_n \triangleq \wp(\mathcal{V}_n)$ as the set of all possible features. $\wp$ is the power set function (*i.e.,* the function that returns the set of all possible subsets). Hence, $\boldsymbol{\chi} \subseteq \mathcal{F}_n$ is an arbitrary set of features. Further, define operator $\texttt{pair} : \boldsymbol{\chi}_1 \to \boldsymbol{\chi}_2$, where $\boldsymbol{\chi}_1, \boldsymbol{\chi}_2 \subseteq \mathcal{F}_n$:

$$\texttt{pair}(\boldsymbol{\chi}) \triangleq \left\{ f \cup g \,\middle|\, f, g \in \boldsymbol{\chi}, f \cup g \notin \boldsymbol{\chi} \right\},$$
$$\texttt{pair}^k(\boldsymbol{\chi}) \triangleq \underbrace{\texttt{pair}(\texttt{pair}(\cdots(\texttt{pair}(\boldsymbol{\chi}))))}_{k \text{ times}},$$

$\texttt{pair}^0(\boldsymbol{\chi}) \triangleq \boldsymbol{\chi}, \texttt{full}(\boldsymbol{\chi}) \triangleq \bigcup_{i=0,\cdots,n} \texttt{pair}^i(\boldsymbol{\chi})$. Essentially, the `pair` operator provides the set of all possible new features built on the top of a given set of features using pairwise conjunction. The `full` operator generates all possible features given a set of features.

Now, given an MDP with a fixed policy, the feature expansion problem can be formulated mathematically. Given $\boldsymbol{\chi}$ as a set of features and the corresponding approximation of the value function under the fixed policy, $\tilde{\boldsymbol{V}}_{\boldsymbol{\chi}} = \boldsymbol{\Phi}_{\boldsymbol{\chi}} \boldsymbol{\theta}$, find $f \in \texttt{pair}(\boldsymbol{\chi})$ that maximizes the following error reduction:

$$ER \quad = \quad \|\boldsymbol{V} - \tilde{\boldsymbol{V}}_{\boldsymbol{\chi}}\| - \|\boldsymbol{V} - \tilde{\boldsymbol{V}}_{\boldsymbol{\chi} \cup \{f\}}\|, \qquad (1)$$

where $||.||$ is the ℓ_2 norm weighted by the steady state distribution. Consequently, all our theoretical analyses are performed in the weighted Euclidean space following the work of [Parr *et al.*, 2007]. The theorem and proof following the next set of assumptions provide an analytical solution that maximizes Equation 1.

Assumptions:
A1. The MDP has a binary d-dimensional state space, $d \in \mathbb{N}^+$ (*i.e.,* $|\mathcal{S}| = 2^d$). Furthermore, each vertex in this binary space corresponds to one unique state; $s \in \{0,1\}^d$.
A2. The agent's policy, π, is fixed.
A3. Each initial feature corresponds to a coordinate of the state space (*i.e.,* $\phi(s) = s$). Hence the number of indices, n, is equal to the number of dimensions, d.

Assumption A1 is a more specific form of a general assumption where each dimension of the MDP can be represented as a finite vector and each dimension has a finite number of possible values. It is simple to verify that such an MDP can be transformed into an MDP with binary dimensions. This can be done by transforming each dimension of the state space with M possible values into M binary dimensions. The MDP with binary dimensions was considered for brevity of the proofs.

Definition The angle between two vectors $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^d, d \in \mathbb{N}^+$ is the smaller angle between the lines formed by the two vectors: $\angle(\boldsymbol{X}, \boldsymbol{Y}) = \arccos\left(\frac{|\langle \boldsymbol{X} \cdot \boldsymbol{Y} \rangle|}{||\boldsymbol{X}||.||\boldsymbol{Y}||}\right)$, where $\langle \cdot, \cdot \rangle$ is the weighted inner product operator. Note that $0 \le \angle(\boldsymbol{X}, \boldsymbol{Y}) \le \frac{\pi}{2}$.

Theorem 3.1 *Given Assumptions A1-A3 and a set of features* $\boldsymbol{\chi}$, *where* $B_n \subseteq \boldsymbol{\chi} \subseteq \mathcal{F}_n$, *then feature* $f^* \in \Omega = \{f | f \in \texttt{pair}(\boldsymbol{\chi}), \angle(\boldsymbol{\phi}_f, \boldsymbol{\delta}) < \arccos(\gamma)\}$ *with the maximum guaranteed error-bound reduction defined in Equation 1 can be calculated as:*

$$f^* = \underset{f \in \Omega}{\operatorname{argmax}} \frac{|\sum_{s \in \mathcal{S}, \phi_f(s)=1} \mathbf{d}(s) \boldsymbol{\delta}(s)|}{\sqrt{\sum_{s \in \mathcal{S}, \phi_f(s)=1} \mathbf{d}(s)}}, \qquad (2)$$

where $\boldsymbol{\delta} = \mathbf{T}(\tilde{\boldsymbol{V}}_{\boldsymbol{\chi}}) - \tilde{\boldsymbol{V}}_{\boldsymbol{\chi}}$ *is the Bellman error vector, and* $\mathbf{d}$ *is the steady state distribution vector.*

The rest of this section provides the building blocks of the proof, followed by a discussion of the theoretical result. Theorem 3.2 states that given an initial set of features, the feature matrix is always full column rank through the process of adding new features using the `pair` operator. Lemma 3.3 provides a geometric property for vectors in d-dimensional space under certain conditions. Theorem 3.4 provides a general guaranteed rate of error-bound reduction when adding arbitrary features to the representation in addition to the convergence proof stated in Theorem 3.6 of [Parr *et al.*, 2007]. Theorem 3.5 narrows down Theorem 3.4 to the case of binary features, where new features are

[2]Note switch in subscript from f (feature) to i (index).

built using the `pair` operator. Finally Theorem 3.1, as stated above, concludes that given the set of potential features obtained by the `pair` operator and filtered based on the their angle with the Bellman error vector; the one with the maximum guaranteed error-bound reduction is identified by Equation 2.

Theorem 3.2 *Given Assumptions A1-A3,* $\forall \boldsymbol{\chi} \subseteq \mathcal{F}_n, \boldsymbol{\Phi}_{\boldsymbol{\chi}}$ *has full rank.*

Proof In appendix.

Insight: Theorem 3.2 shows that the conjunction operator creates a matrix $\boldsymbol{\Phi}_{\mathcal{F}_n}$ that forms a basis for $\mathbb{R}^{|\mathcal{S}|}$ (*i.e.,* $\boldsymbol{\Phi}$ will have $|\mathcal{S}|$ linearly independent columns). The $\boldsymbol{I}$ matrix is another basis for $\mathbb{R}^{|\mathcal{S}|}$, yet no information flows between states (*i.e.,* the coverage of each feature is restricted to one state). When sorting columns of $\boldsymbol{\Phi}_{\mathcal{F}_n}$ based on the size of the features, it starts with features with high coverage (excluding the null feature). As more conjunctions are introduced, the coverage is reduced exponentially (*i.e.,* the number of active features are decreased exponentially by the size of the feature set). Next, we explain how adding binary features can lead to guaranteed approximation error-bound reduction. We begin with a geometric Lemma used in Theorem 3.4.

Lemma 3.3 *Let L be the plane specified by three distinct points* $P, Q, C \in \mathbb{R}^d$*, with* $\alpha = \angle(\boldsymbol{CQ}, \boldsymbol{CP}) > 0$*. Assume that the additional point* $X \in \mathbb{R}^d$ *is not necessarily in L. Define the angles* $\beta = \angle(\boldsymbol{CX}, \boldsymbol{CQ})$ *and* $\omega = \angle(\boldsymbol{CX}, \boldsymbol{CP})$*. Now let* P' *denote the orthogonal projection of P on* $\boldsymbol{CX}$*. If* $\alpha + \beta < \frac{\pi}{2}$*, then* $\|\boldsymbol{PP'}\|$ *is maximized when* $\boldsymbol{CX} \in L$*.*

Proof In Appendix.

We now extend Theorem 3.6 of [Parr *et al.*, 2007] by deriving a lower bound (ζx) on the improvement caused by adding a new feature.

Theorem 3.4 *Given an MDP with a fixed policy, where the value function is approximated as* $\tilde{\boldsymbol{V}}$*, define* $\boldsymbol{\delta} = \mathbf{T}(\tilde{\boldsymbol{V}}) - \tilde{\boldsymbol{V}}$*, and* $\|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| = x > 0$*, where* $\boldsymbol{V}$ *is the optimal value for all states. Then* $\forall \boldsymbol{\phi}_f \in \mathbb{R}^{|\mathcal{S}|} : \beta = \angle(\boldsymbol{\phi}_f, \boldsymbol{\delta}) < \arccos(\gamma)$

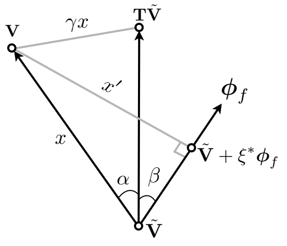
$$\exists \xi \in \mathbb{R} : \|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| - \|\boldsymbol{V} - (\tilde{\boldsymbol{V}} + \xi \boldsymbol{\phi}_f)\| \geq \zeta x, \qquad (3)$$

where γ *is the discount factor and*

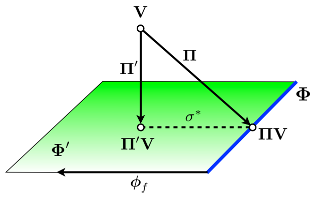
$$\zeta = 1 - \gamma \cos(\beta) - \sqrt{1 - \gamma^2} \sin(\beta) < 1. \qquad (4)$$

Furthermore, if these conditions hold and $\tilde{\boldsymbol{V}} = \boldsymbol{\Phi\theta}$ *with* $\boldsymbol{\phi}_f \notin \mathtt{span}(\boldsymbol{\Phi})$ *then:*

$$\|\boldsymbol{V} - \boldsymbol{\Pi V}\| - \|\boldsymbol{V} - \boldsymbol{\Pi' V}\| \geq \zeta x, \qquad (5)$$



(a)



(b)

Figure 2: a) geometric view of $\boldsymbol{V}, \tilde{\boldsymbol{V}}, \mathbf{T}(\tilde{\boldsymbol{V}})$, and $\boldsymbol{\phi}_f$. As β shrinks x' gets closer to γx. b) increasing the dimensionality of the projection operator.

where $\boldsymbol{\Pi}$ *and* $\boldsymbol{\Pi}'$ *are orthogonal projection operators using* $\boldsymbol{\Phi}$ *and* $\boldsymbol{\Phi}' = [\boldsymbol{\Phi}\ \boldsymbol{\phi}_f]$ *respectively.*

Proof Consider both cases of the orientation of points $\boldsymbol{V}$ and $\mathbf{T}(\tilde{\boldsymbol{V}})$ with respect to each other:

Case $\mathbf{T}(\tilde{\boldsymbol{V}}) \neq \boldsymbol{V}$: Due to the contraction property of the Bellman operator, if $\|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| = x$, then $\|\boldsymbol{V} - \mathbf{T}(\tilde{\boldsymbol{V}})\| \leq \gamma x$. Define α as the $\angle(\boldsymbol{V} - \tilde{\boldsymbol{V}}, \boldsymbol{\delta})$, then using the sine rule:

$$\sin(\alpha) \leq \frac{\|\boldsymbol{V} - \mathbf{T}(\tilde{\boldsymbol{V}})\|}{\|\boldsymbol{V} - \tilde{\boldsymbol{V}}\|} \leq \gamma \quad \Rightarrow \quad \alpha \leq \arcsin(\gamma)$$

Furthermore, by assumption, $0 \leq \beta < \arccos(\gamma) = \pi/2 - \arcsin(\gamma)$. Combined, these conditions indicate that $\alpha + \beta < \pi/2$.

For the next step, given $\xi > 0$, mapping the points $\boldsymbol{V}, \tilde{\boldsymbol{V}}, \mathbf{T}(\tilde{\boldsymbol{V}}), \tilde{\boldsymbol{V}} + \xi\boldsymbol{\phi}_f$ to P, C, Q, X in Lemma 3.3 shows that the orthogonal projection length of vector $\boldsymbol{V} - \tilde{\boldsymbol{V}}$ on $\tilde{\boldsymbol{V}} + \xi\boldsymbol{\phi}_f - \tilde{\boldsymbol{V}}$ is maximized when all four points are coplanar and $\omega = \angle(\boldsymbol{V} - \tilde{\boldsymbol{V}}, \xi\boldsymbol{\phi}_f) = \alpha + \beta$. Notice that the coplanar argument is implicit in the proof of Theorem 3.6 of Parr *et al.* [2007]. Figure 2(a) depicts the geometric view in such a plane, where $\xi^* = \operatorname{argmin}_\xi \|\boldsymbol{V} - (\tilde{\boldsymbol{V}} + \xi\boldsymbol{\phi}_f)\|$, $x' = x\sin(\omega)$.[3] As shown above, $\alpha \leq \arcsin(\gamma)$ and $0 \leq \alpha + \beta < \frac{\pi}{2}$, thus $\sin(\alpha + \beta) \leq \sin(\arcsin\gamma + \beta) = \gamma\cos(\beta) + \sin(\beta)\sqrt{1 - \gamma^2}$, Hence,

$$\begin{aligned} x' &\leq x\left(\gamma\cos(\beta) + \sqrt{1-\gamma^2}\sin(\beta)\right) \\ x - x' &\geq x\left(1 - \gamma\cos(\beta) - \sqrt{1-\gamma^2}\sin(\beta)\right) \equiv \zeta x \end{aligned}$$

Looking at Figure 2(a), it is easy to verify that $x - x' = \|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| - \|\boldsymbol{V} - (\tilde{\boldsymbol{V}} + \xi\boldsymbol{\phi}_f)\|$, which completes the proof for the case $\mathbf{T}(\tilde{\boldsymbol{V}}) \neq \boldsymbol{V}$.

Case $\mathbf{T}(\tilde{\boldsymbol{V}}) = \boldsymbol{V}$: This means that $\alpha = 0$. If $\boldsymbol{\phi}_f$ crosses $\boldsymbol{V}$, it means $\beta = 0$ and $\tilde{\boldsymbol{V}} + \xi^*\boldsymbol{\phi}_f = \boldsymbol{V}$. Hence

[3]Note that ξ can take negative values as well, rendering ϕ_f important only as a line but not as a vector.

$\zeta = 1 - \gamma\cos(\beta) - \sqrt{1-\gamma^2}\sin(\beta) = 1 - \gamma$ and $\|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| - \|\boldsymbol{V} - (\tilde{\boldsymbol{V}} + \xi^* \boldsymbol{\phi}_f)\| = \|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| = x \geq \zeta x$. When $\boldsymbol{\phi_f}$ does not cross $\boldsymbol{V}$, together they form a plane in which $\|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| - \|\boldsymbol{V} - (\tilde{\boldsymbol{V}} + \xi^* \boldsymbol{\phi}_f)\| = x\,(1 - \sin(\beta))$. In order to complete the proof, a lower bound for the above error reduction is derived:

$$0 < \beta < \arccos(\gamma) = {}^{\pi}/_{2} - \arcsin(\gamma), 0 \leq \gamma < 1$$
$$\Rightarrow 0 < \beta + \arcsin(\gamma) < {}^{\pi}/_{2}$$

$$\begin{aligned} \Rightarrow \sin(\beta) &\leq \sin(\beta + \arcsin(\gamma)) \\ &= \gamma\cos(\beta) + \sqrt{1-\gamma^2}\sin(\beta) \end{aligned}$$

$$(1 - \sin(\beta))x \geq (1 - \gamma\cos(\beta) - \sqrt{1-\gamma^2}\sin(\beta))x \equiv \zeta x$$

Now we extend the proof to the linear function approximation case with $\tilde{\boldsymbol{V}} = \boldsymbol{\Phi\theta}$. Since the first part of the proof holds for any approximation, let us consider the case where $\tilde{\boldsymbol{V}} = \boldsymbol{\Pi V}$. Showing $\tilde{\boldsymbol{V}} + \xi^* \boldsymbol{\phi}_f = \boldsymbol{\Pi}'\boldsymbol{V}$ completes the proof as it turns Inequality 3 into Inequality 5. To proceed, we decompose $\boldsymbol{\phi}_f$ into two vectors $\boldsymbol{\phi}_f^{\parallel} \in \mathtt{span}(\boldsymbol{\Phi})$ and $\boldsymbol{\phi}_f^{\perp} \perp \mathtt{span}(\boldsymbol{\Phi})$. First, consider the case where $\boldsymbol{\phi}_f = \boldsymbol{\phi}_f^{\perp}$, Figure 2(b) provides a geometric view of the situation. The blue line and the green plane highlight $\mathtt{span}(\boldsymbol{\Phi})$ and $\mathtt{span}(\boldsymbol{\Phi}')$ respectively. Both $\boldsymbol{\Pi}$ and $\boldsymbol{\Pi}'$ are orthogonal projections into these subspaces. Hence, for any given value function $\boldsymbol{V}$, $\boldsymbol{\Pi}'\boldsymbol{V} = \boldsymbol{\Pi V} + \sigma^* \boldsymbol{\phi}_f$, where, $\sigma^* \triangleq \operatorname{argmin}_\sigma \|(\boldsymbol{V} - \boldsymbol{\Pi V}) - \sigma\boldsymbol{\phi}_f\|$.

The extension to the case where $\boldsymbol{\phi}_f^{\parallel} \neq \boldsymbol{0}$ is straightforward. Consider a subspace defined by two representation matrices $\boldsymbol{\Phi}_1$ and $\boldsymbol{\Phi}_2$ (*i.e.,* $\mathtt{span}(\boldsymbol{\Phi}_1) = \mathtt{span}(\boldsymbol{\Phi}_2)$), and corresponding orthogonal projection operators $\boldsymbol{\Pi}_1$ and $\boldsymbol{\Pi}_2$. Since both operators provide the solution to the same convex optimization (*i.e.,* $\min_{\boldsymbol{\theta}} ||\boldsymbol{V} - \tilde{\boldsymbol{V}}||$), where both domain and target space are identical, their outputs are equal (*i.e.,* $\tilde{\boldsymbol{V}}_1 = \boldsymbol{\Pi}_1\boldsymbol{V} = \boldsymbol{\Pi}_2\boldsymbol{V} = \tilde{\boldsymbol{V}}_2$).[4] Hence if $\boldsymbol{\phi}_f^{\parallel}$ is added as the last column of $\boldsymbol{\Phi}'$, it does not change $\mathtt{span}(\boldsymbol{\Phi}')$ and the result of the projection remains intact. ■

The next theorem specializes the above result to the case of binary features, where new features are built using the conjunction operator.

Theorem 3.5 *Given Assumptions A1-A3,* $\boldsymbol{\chi} \subseteq \mathcal{F}_n, \tilde{\boldsymbol{V}} = \boldsymbol{\Phi_\chi\theta}, \boldsymbol{\delta} = \mathbf{T}(\tilde{\boldsymbol{V}}) - \tilde{\boldsymbol{V}}$, *and* $\|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| = x > 0$, *then* $\forall f \in \mathtt{pair}(\boldsymbol{\chi})$, *if*

$$\eta_f = \frac{|\sum_{s\in\mathcal{S},\phi_f(s)=1} \mathbf{d}(s)\boldsymbol{\delta}(s)|}{\sqrt{\left(\sum_{s\in\mathcal{S},\phi_f(s)=1} \mathbf{d}(s)\right)\left(\sum_{s\in\mathcal{S}} \mathbf{d}(s)\boldsymbol{\delta}^2(s)\right)}} > \gamma$$

$$\begin{aligned} \exists \xi \in \mathbb{R} : \|\boldsymbol{V} - \tilde{\boldsymbol{V}}\| - \|\boldsymbol{V} - (\tilde{\boldsymbol{V}} + \xi\boldsymbol{\phi}_f)\| &\geq \zeta x, &(6) \\ \|\boldsymbol{V} - \Pi\boldsymbol{V}\| - \|\boldsymbol{V} - \Pi'\boldsymbol{V}\| &\geq \zeta x, &(7) \\ \text{where } 1 - \gamma\eta_f - \sqrt{1-\gamma^2}\sqrt{1-\eta_f^2} &= \zeta &(8) \end{aligned}$$

Proof Theorem 3.4 provides a general rate of convergence for the error bound when arbitrary feature vectors are added to the feature matrix. Hence it is sufficient to show that the conditions of Theorem 3.4 holds in this new theorem, namely: 1) $\beta = \angle(\boldsymbol{\phi}_f, \boldsymbol{\delta}) < \arccos(\gamma)$ and 2) $\boldsymbol{\phi}_f \notin \mathtt{span}(\boldsymbol{\Phi_\chi})$. The latter is already shown through Theorem 3.2. As for the former:

$$\begin{aligned} \cos(\beta) &= \frac{|\langle\boldsymbol{\phi}_f \cdot \boldsymbol{\delta}\rangle|}{\|\boldsymbol{\phi}_f\|.\|\boldsymbol{\delta}\|} &(9) \\ &= \frac{|\sum_{s\in\mathcal{S},\phi_f(s)=1} \mathbf{d}(s)\boldsymbol{\delta}(s)|}{\sqrt{\left(\sum_{s\in\mathcal{S},\phi_f(s)=1} \mathbf{d}(s)\right)\left(\sum_{s\in\mathcal{S}} \mathbf{d}(s)\boldsymbol{\delta}^2(s)\right)}}. \end{aligned}$$

Therefore, $\beta = \arccos(\eta_f)$. By the assumption made earlier, $\eta_f > \gamma$. Hence $\beta < \arccos(\gamma)$. Satisfying the preconditions of Theorem 3.4, both Equations 3 and 5 are obtained. Switching $\cos(\beta)$ with η_f in Equation 4 completes the proof. ■

Theorem 3.5 provides sufficient conditions for a guaranteed rate of convergence in the error bound of the value function approximation by adding conjunctions of existing features. It leads directly to Theorem 3.1.

Corollary 3.6 *An algorithm that selects features based on Equation 2, which maximizes Equation 9, is by our definition in Section 2.2 an MP algorithm.*

Insight: Equation 2 shows how feature coverage is a double-edged sword; while greater coverage includes more weighted Bellman error (*i.e.,* the numerator) resulting in a higher convergence rate, it also contributes negatively to the rate of convergence (*i.e.,* the denominator). The ideal feature would be active in a single state with all of the Bellman error. Intuitively, this conclusion is expected, because adding this ideal feature makes the approximation exact. When the weighted sum of Bellman errors is equal for a set of features, the feature with the least coverage is preferable. On the other hand, when all features have the same coverage, the one with the highest weighted Bellman error coverage is ideal. Another interesting observation is the relation between the difficulty of finding features that give the guaranteed error-bound convergence rate and the value of γ. In general, larger values of γ render the MDP harder to solve. Here we can observe the same trend for finding good features as higher values of γ reject more features in the set $\mathtt{pair}(\boldsymbol{\chi})$ due to the constraint $\eta_f > \gamma$. Finally, we note that our theoretical results can be interpreted as

[4]While the corresponding coordinates, $\boldsymbol{\theta}$, in each case can be different, the resulting $\tilde{\boldsymbol{V}}$ are identical.

a mathematical rationale for moving from a coarse to a fine representation, explaining empirical observations in both computer science [Whiteson *et al.*, 2007] and brain/cognitive science [Goodman *et al.*, 2008].

4 Empirical Results

Here we provide experimental evidence of Batch-iFDD's efficiency at policy evaluation, a crucial step in many reinforcement learning algorithms such as Policy Iteration. Policy evaluation is also the traditional setting for comparing feature expansion techniques [*e.g.,* Mannor and Precup, 2006; Painter-Wakefield and Parr, 2012]. We ran our experiments using the RLPy framework, which is available online [Geramifard *et al.*, 2013]. Results are presented in three classical RL domains: Mountain Car, Inverted Pendulum, and System Administrator. The last domain has more than one million states. Our ability to handle such a large domain is a direct consequence of the added efficiency and targeted search in Batch-iFDD.

On each iteration the best weights were found by running LSTD on 10^4 samples gathered using the underlying policy. The A matrix in LSTD was regularized by 10^{-6}. Then the best feature was added to the representation using the corresponding expansion technique. All results were averaged over 30 runs and on each run all algorithms were exposed to the same set of samples. Standard errors are shown as shaded areas highlighting 95% confidence intervals.

We compared two approximations of Equation 2, shown in Equations 10 and 11. The first one comes from our theoretical analysis, where the steady state distribution is approximated by the collected samples. The second one is borrowed from previous work [Geramifard *et al.*, 2011]. In this section, we drop the implicit "Batch-" term and refer to these algorithms as iFDD$^+$ and iFDD[ICML-11] respectively.

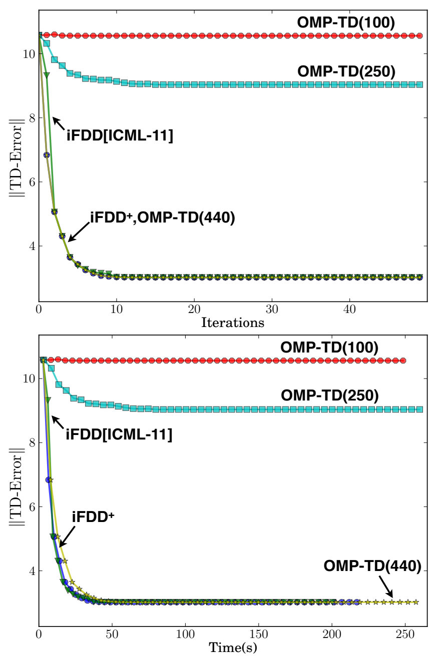
$$\begin{aligned}\delta_i &= r_i + [\gamma\phi(s'_i) - \phi(s_i)]^\top \boldsymbol{\theta}.\\ \tilde{f}_1^* &= \underset{f\in\texttt{pair}(\boldsymbol{\chi})}{\operatorname{argmax}} \frac{|\sum_{i\in\{1,\cdots,m\},\phi_f(s_i)=1}\delta_i|}{\sqrt{\sum_{i\in\{1,\cdots,m\},\phi_f(s_i)=1}1}}, \quad (10)\\ \tilde{f}_2^* &= \underset{f\in\texttt{pair}(\boldsymbol{\chi})}{\operatorname{argmax}} \sum_{i\in\{1,\ldots,m\},\phi_f(s_i)=1} |\delta_i| \quad (11)\end{aligned}$$

We also implemented and compared against variants of OMP-TD [Painter-Wakefield and Parr, 2012], the previous state of the art MP algorithm in RL. Since this algorithm requires a set of potential features at initialization, we tested several different sizes of potential feature sets, each built by including features in increasingly finer levels of the concept lattice until the cap was reached. Note that in two dimensional problems (where only one layer of conjunction is available), if the OMP-TD potential feature pool contains every possible feature then the results of running OMP-TD and iFDD$^+$ will be identical, since both algorithms run the same optimization on the same set of features.

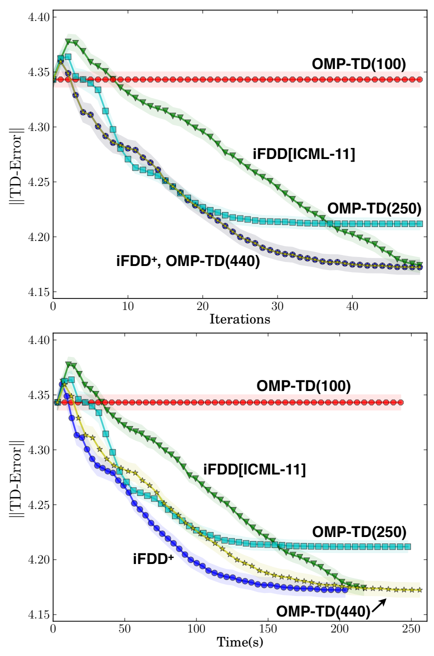
The first domain was Mountain Car [Sutton and Barto, 1998], with base features defined as discretizations of position and velocity into 20 partitions each, leading to 40 base features. The policy evaluated was to accelerate in the direction of the car's current velocity. Figure 3(a)-top shows the $\|\text{TD-Error}\|_2$ over the sample set for the iFDD and OMP-TD methods versus the number of feature-generating iterations. Since all the techniques start with the same set of features, their errors are identical at iteration zero. OMP-TD with pool sizes up to 250 did not capture the value function well. With $440 = 20 \times 20 + 40$ potential features, OMP-TD had access to all possible features, and as predicted it performed identically to iFDD$^+$. iFDD[ICML-11] performed similar to the best results. Figure 3(a)-bottom depicts the same results based on the wall-clock time. The iFDD techniques were at least 30 seconds faster than the OMP-TD methods as they considered fewer features on each iteration.

Next we considered the classical Inverted Pendulum domain [See Lagoudakis and Parr, 2003]. The feature setting was identical to the Mountain Car problem. The fixed policy pushed the pendulum in the opposite direction of its angular velocity. TD-error results as described before are presented for this domain in Figure 3(b). Again OMP-TD with 100 features did not have access to the necessary features and performed poorly. With 250 features, OMP-TD performed much better, but converged to a relatively less accurate representation after about 25 iterations. With access to the full set of features, OMP-TD(440) mirrored the performance of iFDD$^+$, both achieving the best results. In this domain, the less accurate approximation of Equation 2 used by iFDD[ICML-11] caused a significant drop in its performance compared to iFDD$^+$. However, it eventually exceeded the performance of OMP-TD(250) and caught up to iFDD$^+$ by expanding important features. There is a small initial rise in error for most of the algorithms, which we believe is due to the use of regularization. In terms of computational complexity, we see the same pattern as in the Mountain Car domain, where the iFDD methods were computationally more efficient (about 20% faster) than the OMP-TD techniques.

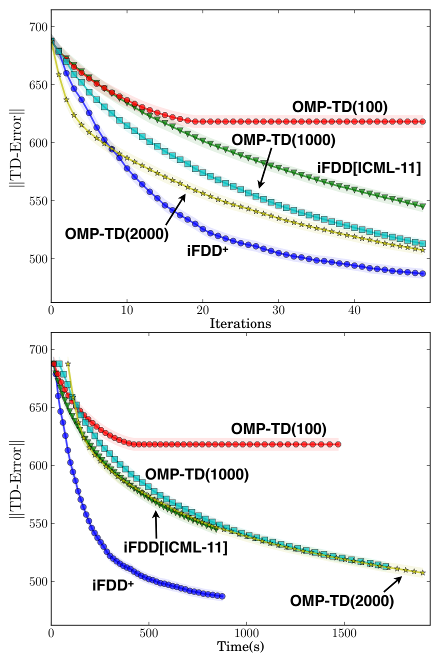
The third experiment considered the System Administrator domain [Guestrin *et al.*, 2001] with 20 computers and a fixed network topology. Each computer can either be up or down (following our example in Section 2.3), so there were 40 base features. The size of the state space is 2^{20}. Computers go up or



(a) MountainCar



(b) Inverted Pendulum



(c) System Administrator

Figure 3: Empirical results from the a) Mountain Car, b) Inverted Pendulum, and c) System Administrator domains. The Y-axis depicts the ℓ_2 norm of the TD prediction error plotted against (top) the number of feature expansions and (bottom) wall clock time. The colored shaded areas highlight the 95% confidence intervals.

down depending on the status of their neighbors [See Guestrin *et al.*, 2001]. Allowed actions are to reboot one machine at each timestep or do nothing: in our case the policy uniformly randomly rebooted one of the down machines. Results are shown in Figure 3(c). The general theme remains the same, except for three observations: 1) for the first eight iterations, OMP-TD(2000) outperformed iFDD$^+$, 2) after 10 iterations iFDD$^+$ outperformed all of the OMP-TD techniques with pool sizes up to 2000, and 3) the speed advantage of the iFDD techniques was much more prominent. Based on clock time iFDD[ICML-11] was comparable to the best OMP-TD technique, while iFDD$^+$ achieved the final performance of OMP-TD(2000) more than 4 times faster. The reason for the initial OMP-TD success is that it was able to add complex features (conjunctions of several terms) early on without adding the coarser (subsumed) conjunctions that iFDD adds first.[5] The second observation is explained by the fact that iFDD$^+$ expands the set of potential features in a guided way, allowing it to discover crucial fine-grained features. Specifically iFDD$^+$ discovered features with 8 terms. Finally, as the size of the potential feature pool grew, the OMP-TD techniques required significantly more computation time. iFDD techniques on the other hand, only considered possible new pair-wise features, and scaled much better to larger MDPs.

5 Conclusions

We introduced Batch-iFDD (and Batch-iFDD$^+$) for feature construction in an RL setting and proved that it is a Matching Pursuit algorithm. Unlike previous MP techniques, Batch-iFDD expands the pool of potential features incrementally, hence searching the concept lattice more efficiently than previous MP techniques. Our empirical results support this finding as Batch-iFDD$^+$ outperformed the previous state of the art MP algorithm in three benchmarks, including a domain with over one million states.

It should be noted, that OMP-TD techniques are more general than Batch-iFDD techniques as they can work with arbitrary feature functions rather than binary functions. Also, Batch-iFDD is not immune to the poor selection of base features. For instance, in continuous state spaces base features can be built by discretizing each dimension using the indicator function, yet finding the "right" discretization for high dimensional problems can be challenging.

Beyond the results of this paper, Equation 2 provides insight as to why it is beneficial to add coarse features in the early stages of learning and finer features later on. In the early stages of learning, when feature

[5]The feature pool for OMP-TD(2000) consisted of 760 and 1,200 features with 2 and 3 terms respectively.

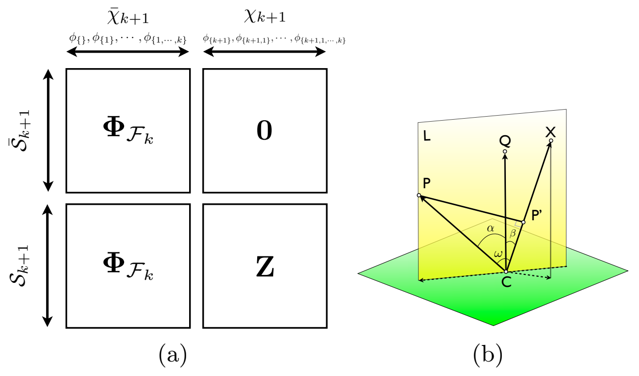

$\bar{\chi}_{k+1}$
$\phi_{\{\}}, \phi_{\{1\}}, \cdots, \phi_{\{1,\cdots,k\}}$
χ_{k+1}
$\phi_{\{k+1\}}, \phi_{\{k+1,1\}}, \cdots, \phi_{\{k+1,1,\cdots,k\}}$
$\bar{\mathcal{S}}_{k+1}$
$\mathcal{S}_{k+1}$
$\mathbf{\Phi}_{\mathcal{F}_k}$
$\mathbf{0}$
$\mathbf{\Phi}_{\mathcal{F}_k}$
$\mathbf{Z}$
(a)
L
Q
X
P
P'
α
β
ω
C
(b)

Figure 4: a) Depiction of $\mathbf{\Phi}_{\mathcal{F}_{k+1}}$ using $\mathbf{\Phi}_{\mathcal{F}_k}$ as the building block. Note that features are not sorted based on their cardinality order, but it does not change the rank of the resulting matrix. b) A $3D$ visualization of d dimensional points L and Q and vector X with C as the center. $\|\boldsymbol{PP'}\|$ is maximized when $\omega = \alpha + \beta$.

weights have not been adjusted, the Bellman error is generally large everywhere. Therefore coarse features with large coverage have higher chances of having good error-bound convergence rates due to the numerator of Equation 2. As weights are updated, the Bellman error is reduced correspondingly. The reduced Bellman error will make the denominator of Equation 2 the deciding factor, rendering coarse features with large coverage ineffective. This transition may partially explain empirical results on RL agents exploring autonomously [Whiteson *et al.*, 2007] and human subjects performing classification [Goodman *et al.*, 2008], where both benefited from using coarse features at the beginning of learning but then progressed to finer-grained features to make better sense of a complex world.

Acknowledgments

We thank Ronald Parr and our anonymous reviewers for insightful feedback. This research was sponsored by ONR grants N00014-07-1-0749 and N00014-11-1-0688.

A Proof of Theorem 3.2

Lemma A.1 *Given $m, n \in \mathbb{N}^+$ and $m \le n$, if $\mathbf{X}_{m\times n}$ and $\mathbf{Z}_{m\times n}$ are full column rank matrices with real elements and $\mathbf{Y}_{m\times n}$ is arbitrary matrix, then matrix* $\begin{bmatrix} \mathbf{X} & \mathbf{0} \\ \mathbf{Y} & \mathbf{Z} \end{bmatrix}$ *is a full column rank matrix.*

Proof The proof follows from the definition of the matrix as both $\mathbf{X}$ and $\mathbf{Z}$ have linear independent columns.

Theorem A.2 *Given Assumptions A1-A3, $\forall n \in \mathbb{N}^+, \mathbf{\Phi}_{\mathcal{F}_n}$ is invertible.*

Proof First note that $\mathbf{\Phi}_{\mathcal{F}_n}$ is a square matrix as $|\mathcal{F}_n| = |\mathcal{S}| = 2^n$. Hence it is sufficient to show that $\mathbf{\Phi}_{\mathcal{F}_n}$ has independent columns. The rest of the proof is through induction on n:

$(n = 1)$: The MDP has two states. Hence $\mathbf{\Phi}_{\mathcal{F}_1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $det(\mathbf{\Phi}_{\mathcal{F}_1}) = 1$. Notice that the first column corresponds to the null feature (*i.e.,* $\{\}$) which returns 1 for the single state with no active features.

$(n = k)$: Assume that $\mathbf{\Phi}_{\mathcal{F}_k}$ has independent columns.

$(n = k+1)$: Based on the previous assumption, $\mathbf{\Phi}_{\mathcal{F}_k}$ has linearly independent columns. Hence it is sufficient to show that $\mathbf{\Phi}_{\mathcal{F}_{k+1}}$ can be written as $\begin{bmatrix} \mathbf{X} & \mathbf{0} \\ \mathbf{Y} & \mathbf{Z} \end{bmatrix}$, where $\mathbf{X} = \mathbf{Y} = \mathbf{\Phi}_{\mathcal{F}_k}$, and $\mathbf{Z}$ has linearly independent columns. Lemma A.1 then completes the proof.

The new added dimension, $k+1$, doubles the number of states because $|\mathcal{S}| = 2^{k+1}$. The new dimension also doubles the total number of possible features, as for any given set with size k, the total number of its subsets is 2^k. Divide states into the two following sets:

$$\mathcal{S}_{k+1} = \{s|\phi_{\{k+1\}}(s) = 1\}, \bar{\mathcal{S}}_{k+1} = \{s|\phi_{\{k+1\}}(s) = 0\}.$$

Similarly, divide features into two sets:

$$\chi_{k+1} = \{f|f \in \mathcal{F}_{k+1}, k+1 \in f\},$$
$$\bar{\chi}_{k+1} = \{f|f \in \mathcal{F}_{k+1}, k+1 \notin f\}.$$

Construct rows and columns of $\mathbf{\Phi}_{\mathcal{F}_{k+1}}$, following Figure 4. The values of the top left and bottom left of the matrix are $\mathbf{\Phi}_{\mathcal{F}_k}$, and the value of the top right of the matrix is $\mathbf{0}$. As for the bottom right ($\mathbf{Z}$), note that for all the corresponding states, $\phi_{\{k+1\}}(s) = 1$. Hence,

$$(\phi_{\{k+1\}}(s), \phi_{\{k+1,1\}}(s), \cdots, \phi_{\{k+1,1,\cdots,k\}}(s))$$
$$= (1, \phi_{\{1\}}(s), \cdots, \phi_{\{1,\cdots,k\}}(s)).$$

We know from the induction assumption that except for the first column, all other columns are linearly independent. Finally observe that the first column is the only column within $\mathbf{Z}$, with a 1 corresponding to the state with no active features and is independent of all other columns. ∎

Theorem 3.2 follows from Theorem A.2.

B Proof of Lemma 3.3

Proof Let us first assume that $\boldsymbol{CX} \notin L$, hence there exists a three dimensional subspace defined by $\boldsymbol{CX}$ and L. Figure 4 depicts such a space. Then, $\text{argmax}_\omega \|\boldsymbol{PP'}\| = \text{argmax}_\omega \|\boldsymbol{CP}\| \sin(\omega) = \text{argmax}_\omega \sin(\omega)$. Since $0 < |\alpha - \beta| \le \omega \le \alpha + \beta < \frac{\pi}{2}$, then $\text{argmax}_\omega \|\boldsymbol{PP'}\| = \alpha + \beta$, which implies that $\boldsymbol{CX} \in L$ and thus is a contradiction. ∎

References

Steven J. Bradtke and Andrew G. Barto. Linear least-squares algorithms for temporal difference learning. *Journal of Machine Learning Research (JMLR)*, 22:33–57, 1996.

Alborz Geramifard, Finale Doshi, Joshua Redding, Nicholas Roy, and Jonathan How. Online discovery of feature dependencies. In Lise Getoor and Tobias Scheffer, editors, *International Conference on Machine Learning (ICML)*, pages 881–888. ACM, June 2011.

Alborz Geramifard, Robert H Klein, and Jonathan P How. RLPy: The Reinforcement Learning Library for Education and Research. `http://acl.mit.edu/RLPy`, April 2013.

Noah D. Goodman, Joshua B. Tenenbaum, Thomas L. Griffiths, and Jacob Feldman. Compositionality in rational analysis: Grammar-based induction for concept learning. In M. Oaksford and N. Chater, editors, *The probabilistic mind: Prospects for Bayesian cognitive science*, 2008.

Carlos Guestrin, Daphne Koller, and Ronald Parr. Max-norm projections for factored mdps. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 673–682, 2001.

Michail G. Lagoudakis and Ronald Parr. Least-squares policy iteration. *Journal of Machine Learning Research (JMLR)*, 4:1107–1149, 2003.

Stephen Lin and Robert Wright. Evolutionary tile coding: An automated state abstraction algorithm for reinforcement learning. In *AAAI Workshop: Abstraction, Reformulation, and Approximation*, Atlanta, Georgia, USA, 2010.

Sridhar Mahadevan, Mauro Maggioni, and Carlos Guestrin. Proto-value functions: A Laplacian framework for learning representation and control in Markov decision processes. *Journal of Machine Learning Research (JMLR)*, 8:2007, 2006.

Shie Mannor and Doina Precup. Automatic basis function construction for approximate dynamic programming and reinforcement learning. In *International Conference on Machine Learning (ICML)*, pages 449–456. ACM Press, 2006.

Christopher Painter-Wakefield and Ronald Parr. Greedy algorithms for sparse reinforcement learning. In *International Conference on Machine Learning (ICML)*, pages 968–975. ACM, 2012.

Ronald Parr, Christopher Painter-Wakefield, Lihong Li, and Michael Littman. Analyzing feature generation for value-function approximation. In *International Conference on Machine Learning (ICML)*, pages 737–744, New York, NY, USA, 2007. ACM.

Bohdana Ratitch and Doina Precup. Sparse distributed memories for on-line value-based reinforcement learning. In *European Conference on Machine Learning (ECML)*, pages 347–358, 2004.

David Silver, Richard S. Sutton, and Martin Müller. Sample-based learning and search with permanent and transient memories. In *International Conference on Machine Learning (ICML)*, pages 968–975, New York, NY, USA, 2008. ACM.

Peter Stone, Richard S. Sutton, and Gregory Kuhlmann. Reinforcement learning for RoboCup-soccer keepaway. *International Society for Adaptive Behavior*, 13(3):165–188, 2005.

Nathan R. Sturtevant and Adam M. White. Feature construction for reinforcement learning in hearts. In *5th International Conference on Computers and Games*, 2006.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, 1998.

Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

Shimon Whiteson, Matthew E. Taylor, and Peter Stone. Adaptive tile coding for value function approximation. Technical Report AI-TR-07-339, University of Texas at Austin, 2007.

Structured Message Passing

Vibhav Gogate
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, USA.
vgogate@hlt.utdallas.edu

Pedro Domingos
Computer Science & Engineering
University of Washington
Seattle, WA 98195, USA.
pedrod@cs.washington.edu

Abstract

In this paper, we present structured message passing (SMP), a unifying framework for approximate inference algorithms that take advantage of structured representations such as algebraic decision diagrams and sparse hash tables. These representations can yield significant time and space savings over the conventional tabular representation when the message has several identical values (context-specific independence) or zeros (determinism) or both in its range. Therefore, in order to fully exploit the power of structured representations, we propose to artificially introduce context-specific independence and determinism in the messages. This yields a new class of powerful approximate inference algorithms which includes popular algorithms such as cluster-graph Belief propagation (BP), expectation propagation and particle BP as special cases. We show that our new algorithms introduce several interesting bias-variance trade-offs. We evaluate these trade-offs empirically and demonstrate that our new algorithms are more accurate and scalable than state-of-the-art techniques.

1 INTRODUCTION

Access to fast, scalable and accurate approximate inference algorithms is the key to the successful application of graphical models to real world problems. As a result, several approximate inference algorithms have been proposed to date, in a large body of literature spanning several decades. Existing algorithms can be classified into two broad types: message passing based and sampling or simulation based. Message passing algorithms operate by passing messages over the edges of a cluster graph derived from the graphical model while sampling algorithms operate by randomly generating variable configurations. In this paper, we focus on message passing algorithms and propose a new framework, *structured message passing* (SMP) which provides a principled approach for taking advantage of structured approaches for representing and manipulating messages.

We propose SMP because popular, approximate message passing algorithms such as belief propagation (BP) [22], its various generalizations [19, 34], and expectation propagation (EP) [20, 21] rely on tabular representations. Tabular representations, although easy to use and manipulate, can be exponentially worse in terms of size and processing time than structured approaches such as algebraic decision diagrams (ADDs) [1] and sparse hash tables. As a result, in presence of time and space resource constraints, which is often the case in practice, we are unable to apply several more efficient and potentially more accurate classes of algorithms to real-world problems.

Over the last decade, there has been much research on developing exact inference algorithms that exploit the power of structured representations. Notable examples are Cachet [25], ACE [4], and ADD-based variable elimination [3]. The first two use weighted propositional features for representing messages while ADD-based variable elimination uses ADDs [1]. By taking advantage of structural features such as context-specific independence (CSI) [2] and determinism, the aforementioned algorithms can solve much larger problems than the junction tree algorithm [17]. For approximate inference, however, structured representations have not been investigated as much (cf. [5, 10, 18, 27]) and their power has not been fully realized.

The basic idea in SMP is quite simple. Unlike BP and EP in which we associate each cluster and each edge in a cluster graph with a single tabular function and a product of tabular functions respectively [33], in SMP we associate a structured representation of a function with each cluster and each edge, yielding a *structured cluster graph*. We assume that the structured representation not only defines a suitable (computer) representation but also various inference operators such as sum and product. Thus, given a cluster graph and a message passing schedule, each representation defines a structured message passing algorithm.

We show that in spite of its simplicity, SMP enables us to define more powerful BP and EP algorithms as well as several new classes of (principled) message passing algorithms. In particular, when the inference operators are lossless, i.e., they faithfully represent the message, we get the cluster graph BP algorithm. When the inference operators are lossy and minimize the KL divergence between the original function and the lossy representation, we get the EP algorithm. Defining new lossy operators yields new classes of algorithms. However, since the structured representations can be exponentially more efficient than the tabular representation, the resulting SMP algorithms are likely to be much more efficient in terms of time and space complexity. Thus, given a bound on time and space complexity, SMP will allow much larger clusters than tabular BP and EP. Since the accuracy typically increases with the cluster size, it is likely that SMP algorithms will be more accurate than tabular BP and EP.

We consider a possible instance of the class of SMP algorithms in which we artificially introduce determinism and CSI in the messages. Such messages can then be efficiently represented using structured approaches, yielding a significant reduction in complexity. Moreover, if each new message includes assignments that have relatively high information content, the resulting algorithm will also have high accuracy. We propose to introduce determinism via Monte Carlo simulation (e.g., via Gibbs sampling or importance sampling), retaining only the sampled (and therefore potentially high-probability) partial assignments within each cluster. Following [10, 30], we propose to introduce CSI by quantizing messages, namely reducing the number of distinct values in the range of the message by replacing a number of values that are close to each other by a single value.

We show that our new SMP algorithm introduces several bias-variance trade-offs. Specifically, we show that given a set of samples and a fixed error bound for quantization, increasing the cluster size increases the variance but reduces the bias. On the other hand, for a fixed error bound and cluster size, increasing the sample size decreases the variance and therefore improves accuracy.

Within our algorithm and the SMP framework, we consider two structured representations: sparse hash tables and algebraic decision diagrams,[1] define lossy and lossless operators for them and empirically evaluate their efficacy on various benchmarks. For comparison, we use iterative join graph propagation (IJGP) [19], co-winner of 2010 UAI competition [8], and evaluate our algorithms on the task of computing all single variable marginal distributions. Our experiments show that our new algorithm is superior to IJGP.

The rest of the paper is organized as follows. In section 2, we describe notation and preliminaries. In section 3, we introduce structured cluster graphs and describe structured representations and operators in section 4. In section 5, we present our new algorithm that is a possible instance of SMP and analyze bias-variance tradeoffs for it. We empirically evaluate our new algorithm in section 6. We discuss related work in section 7 and conclude in section 8.

2 PRELIMINARIES AND NOTATION

A (discrete) graphical model or a Markov network (cf. [6, 15, 24]), denoted by $\mathcal{G}$, is a triple $(\mathbf{X}, \mathbf{D}, \Phi)$, where $\mathbf{X} = \{X_1, \ldots, X_n\}$ is a set of variables, $\mathbf{D} = \{D(X_1), \ldots, D(X_n)\}$ is a set of domains of variables, where $D(X_i)$ is the domain of X_i and $\Phi = \{\phi_1, \ldots, \phi_m\}$ is a set of functions (also called factors or potentials). Each function ϕ_i is defined over a subset of variables, called its scope, denoted by $S(\phi_i)$. Let $D(\mathbf{X})$ denote the Cartesian product of the domains of all variables in $\mathbf{X}$. Let $\mathbf{x} = (x_1, \ldots, x_n) \in D(\mathbf{X})$ where $x_i \in D(X_i)$ denote an assignment of values to all variables in $\mathbf{X}$. A Markov network represents the following probability distribution.

$$P_{\mathcal{G}}(\mathbf{x}) = \frac{\prod_{i=1}^{m} \phi_i(\mathbf{x}_{S(\phi_i)})}{\sum_{\mathbf{x} \in D(\mathbf{X})} \prod_{i=1}^{m} \phi_i(\mathbf{x}_{S(\phi_i)})} \tag{1}$$

where $\mathbf{x}_{S(\phi_i)}$ is the projection of $\mathbf{x}$ on $S(\phi_i)$. We will often abuse notation and write $\phi_i(\mathbf{x}_{S(\phi_i)})$ as $\phi_i(\mathbf{x})$. The denominator of Eq. (1) is a normalization constant, called the *partition function*. Common queries over graphical models are computing the partition function and the marginal distribution $P_{\mathcal{G}}(X_i)$ for all variables $X_i \in \mathbf{X}$.

Cluster graph belief propagation (BP) is an approximate message passing algorithm for computing variable marginals. It operates on a data structure called the cluster graph defined below:

Definition 1. Given a graphical model $\mathcal{G} = (\mathbf{X}, \mathbf{D}, \Phi)$, a **cluster graph** is a graph $G(\mathbf{V}, \mathbf{E})$ in which each vertex $V \in \mathbf{V}$ and edge $E \in \mathbf{E}$ is associated with a subset of variables, denoted by $L(V)$ and $L(E)$ respectively such that: (i) for every function $\phi \in \Phi$, there exists a vertex $L(V)$ such that $S(\phi) \subseteq L(V)$; and (ii) for every variable $X \in \mathbf{X}$, the set of vertices and edges in G that mention X form a connected sub-tree of G (*running intersection property*).

In cluster graph BP, we first put each function $\phi \in \Phi$ in a cluster V such that $S(\phi) \subseteq L(V)$. Then each node V_i sends the following message to a node V_j on the edge $E_{i,j}$, iteratively until convergence

$$m_{i\to j}(\mathbf{y}) = \sum_{\mathbf{z}} \prod_{\phi \in \Phi(V_i)} \phi(\mathbf{y}, \mathbf{z}) \prod_{V_k \in N(i,j) \setminus \{V_j\}} m_{k \to i}(\mathbf{y}, \mathbf{z})$$

[1]Note that SMP is a general approach for easily designing message-passing algorithms and as such can be used with any structured representation, not just ADDs and sparse hash tables. For example, it is relatively straight-forward to extend our basic framework to Affine ADDs [26].

where $\mathbf{Y} = L(E_{i,j})$, $\mathbf{y} \in D(\mathbf{Y})$, $\mathbf{Z} = L(V_i) \setminus L(E_{i,j})$, $\mathbf{z} \in D(\mathbf{Z})$, $\Phi(V_i)$ is the set of functions from the graphical model assigned to V_i, $m_{i \to j}$ is the message sent from V_i to V_j, and $N(i,j)$ is the set of neighbors of V_i in G. Once the messages have converged, we can recover the marginal distribution for any variable $X_i \in \mathbf{X}$ by finding a cluster $V \in \mathbf{V}$ that mentions X_i, multiplying all functions and incoming messages to the cluster and then summing out all variables other than X_i from the resulting function. The cluster graph BP algorithm may not converge. In such cases, we can put a bound on the number of iterations and stop the algorithm once it exceeds this bound.

The message passing approach described above is called sum-product message passing. An alternative approach, which has smaller time complexity but higher space complexity is belief-update message passing (see [15] for more details). When the cluster graph is a tree, cluster graph BP is exact and coincides with the junction tree algorithm. The time and space complexity of cluster graph BP is $O(|\mathbf{V}| \exp(\max_{V \in \mathbf{V}} |L(V)|))$ assuming that each message is represented using a table.

The expectation propagation algorithm (EP) [20] operates on a cluster graph by passing approximate messages. In EP, we associate a product of functions with each edge [33], denoted by $\widetilde{m}_{i \to j} = \prod_k \widetilde{m}_{i \to j,k}$. The main idea here is to approximate a large message which is computationally infeasible using a tractable message $\widetilde{m}_{i \to j}$ such that the KL divergence[2] between the two is minimized.

Unlike cluster graph BP, in EP, sum-product and belief-update message passing will yield different estimates. Often, however, we prefer belief-update message passing in EP because it yields more accurate answers in practice.

3 STRUCTURED CLUSTER GRAPHS

In a cluster graph, each cluster and each edge is associated with a tabular function or a product of tabular functions. The main, fairly simple idea in structured cluster graphs is to associate each edge and each cluster with a parametric (i.e., structured) representation of a function. The parametric representation of a function is a pair $(\mathcal{R}, \mathbf{w})$ where $\mathcal{R}$ denotes the structure and $\mathbf{w}$ is a set of real-valued parameters. We assume throughout that $\mathcal{R}$ determines the complexity of representing the function. We also assume that the structure is fixed or we have bound on its complexity.

Definition 2. Given a graphical model $(\mathbf{X}, \mathbf{D}, \Phi)$, a **structured cluster graph** (SCG) is a graph $G(\mathbf{V}, \mathbf{E})$ in which each vertex $V \in \mathbf{V}$ and each edge $E \in \mathbf{E}$ is associated with a parametric representation of a function, denoted by $\mathcal{R}_V$ and $\mathcal{R}_E$ respectively, such that: (i) for every function $\phi \in \Phi$, there exists a vertex V such that $S(\phi) \subseteq S(\mathcal{R}_V)$ and (ii) for every variable $X \in \mathbf{X}$, the set of vertices and edges in G that mention X form a connected sub-tree of G.

To perform message passing over a SCG, we need to define the sum, product and division operators over the parametric representation. We assume that the *representation system* used defines these operators for us. In addition, we assume that the system provides a *projection* operator, which takes a function ϕ and a parametric representation $(\mathcal{R}, \mathbf{w})$ as input and sets the parameters $\mathbf{w}$. In other words, the projection operator yields an instantiation of $(\mathcal{R}, \mathbf{w})$, which we will denote by $\mathcal{R}[\phi]$. We say that $\mathcal{R}[\phi]$ is *lossless* if we can recover $\phi(\mathbf{y})$ for all $\mathbf{y} \in D(S(\phi))$, i.e., $\mathcal{R}[\phi](\mathbf{y}) = \phi(\mathbf{y})$. Otherwise, it is *lossy*.

The *product* and *division* operators take two instantiations $\mathcal{R}_A[\phi_i]$ and $\mathcal{R}_B[\phi_j]$, and a target representation $\mathcal{R}_C$ as input, and output $\mathcal{R}_C[\phi_k]$ where $\phi_k = \phi_i.\phi_j$ and $\phi_k = \phi_i/\phi_j$ respectively. The *sum* operator takes as input an instantiation $\mathcal{R}_A[\phi_i]$, a representation $\mathcal{R}_B$, and a set of variables $\mathbf{Y} \subseteq S(\phi_i)$ as input and outputs $\mathcal{R}_B[\phi_j]$ where $\phi_j = \sum_{\mathbf{Y}} \phi_i$. We say that the sum, product and division operators are lossless if their output is lossless. Otherwise, they are lossy. The lossy sum, product and division operators can be defined in terms of their lossless analogues using the projection operator; the lossy instantiation $\mathcal{R}_{LY}[\phi]$, is simply a projection of the lossless instantiation $\mathcal{R}_{LS}[\phi]$, on $\mathcal{R}_{LY}$.

The message passing algorithm over a SCG, which we will refer to as *structured message passing* (SMP) operates as follows. First, we initialize the SCG by initializing the parametric representation at each edge and each cluster to the uniform distribution (or to some other distribution based on prior knowledge). Then, for each function $\phi \in \Phi$, we select a cluster V such that $S(\phi) \subseteq L(V)$ and multiply $\mathcal{R}_V[\phi]$ with the current structured representation, say $\mathcal{R}_V[\phi_V]$ at V, storing the result in $\mathcal{R}_V[\phi_V]$. Then, we pass messages, as usual, between the clusters, using the sum, division and product operators, until convergence. In sum-product propagation only the sum and product operators are used while in the belief update propagation all the three operators are used. The complexity of structured message passing is dependent on the representation system used.

It is straight-forward to show that:

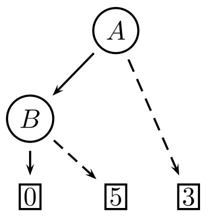
Proposition 1. *SMP is equivalent to the cluster graph BP algorithm if all operators are lossless. Similarly, SMP is equivalent to the EP algorithm under the restriction that the sum, product and division operators are lossy and all messages on all edges $E_{i,j} = (V_i, V_j)$ are such that they minimize the K-L divergence between the actual message $m_{i \to j}$ and the represented message $\mathcal{R}_{E_{i,j}}[m_{i \to j}]$.*

In spite of this equivalence, note that SMP with the afore-

[2]KL divergence between two distributions P and Q is given by $\sum_{\mathbf{x}} P(\mathbf{x}) \log(P(\mathbf{x})/Q(\mathbf{x}))$. To compute KL divergence between two functions, we normalize the functions and then compute KL divergence between them.

A	B	$\phi(a,b)$
0	0	3
0	1	3
1	0	5
1	1	0

(a)



(b)

Figure 1: (a) Tabular representation of a Boolean function, (b) Its ADD representation. In the ADD representation the solid and dashed arcs correspond to the true and false assignments of the parent variable respectively.

mentioned restrictions is more powerful than both tabular BP and tabular EP because by using structured representations that are more efficient than tabular representations, in practice, we can run SMP on graphs having much larger cluster size than both BP and EP. As the accuracy typically increases with the cluster size, we expect SMP to be more accurate than tabular EP and BP algorithms having comparable computational complexity.

4 STRUCTURED REPRESENTATIONS

In this section, we consider two structured representations, sparse tables and algebraic decision diagrams, and describe lossless sum, product, division and assignment operators for them. Then, we show how we can exploit the power of these representations by defining lossy operators.

Sparse Tables Sparse or zero-suppressed tables are often useful when a substantial number of zeros are present in the graphical model [13, 16]. Instead of storing a real number for all possible configurations of variables, the function is represented as a list of tuples having non-zero values. For example, the sparse representation of the tabular representation given in Fig. 1(a) is a table that contains only the first three entries. The non-zero tuples are typically stored in a hash table for fast access. It is easy to define lossless sum, product, division and assignment operators over this representation. For instance, the product operator corresponds to database hash join and the sum operator corresponds to database project (cf. [28]). The complexity of these operations is linear in the size of the input and output tables, assuming constant time table lookup. The lossless operators define a cluster graph BP algorithm, which is likely to be more efficient than the tabular approach when determinism is present. However, when no determinism is present, the resulting BP algorithm will be slightly inefficient than the tabular approach because of the constant time overhead introduced by the sparse operators.

Algebraic Decision Diagrams Algebraic decision diagrams (ADDs) [1] are an efficient representation of real-valued Boolean functions having many identical values in their range. ADDs are directed acyclic graphs (DAG) and have two types of nodes: leaf nodes which are labeled by real-values and non-leaf nodes which are labeled by variables. Each decision node has two outgoing arcs corresponding to the true and false assignments of the corresponding variable. ADDs enforce a strict variable ordering from the root to the leaf node and impose the following three constraints on the DAG: (i) no two arcs emanating from a decision node can point to the same node, (ii) if two decision nodes have the same variable label, then they cannot have (both) the same true child node and the same false child node and (iii) no two leaf nodes are labeled by the same real value. ADDs that do not satisfy these constraints are referred to as unreduced ADDs while those that do are called reduced ADDs. An unreduced ADD can be reduced by merging isomorphic subgraphs and eliminating any nodes whose two children are isomorphic (see [1] for more details). A reduced, ordered ADD is a canonical representation. Namely, two functions will have the same ADD (under the same variable ordering) iff they are the same. For example, Fig. 1(b) shows the ADD representation of the function given in Fig. 1(a).

It is easy to define lossless sum, product and division operators using standard ADD operations (and in practice, implement them using open-source ADD packages such as CUDD [29]). The complexity of these operations is linear in the size of the largest ADD. Note that any non-Boolean function can be converted to a Boolean function by introducing a Boolean variable for each variable-value pair and adding Boolean constraints which ensure that each variable is assigned exactly one value (cf. [32]). Therefore, our approach is also applicable to multi-valued variables.

ADDs and Sparse Tables as Features ADDs and sparse tables can be interpreted as representations of weighted features (or propositional formulas) defined over their variables. Each entry in the sparse table represents a simple conjunctive feature while each leaf node of an ADD represents a complex feature that is a disjunction of several conjunctive features. For example, the first two entries in the sparse table in Fig. 1(a) represent the conjunctive weighted features $[(\neg A \wedge \neg B), 3]$ and $[(\neg A \wedge B), 3]$ respectively while the rightmost leaf node in the ADD in Fig. 1(b) represents the complex weighted feature $[((\neg A \wedge \neg B) \vee (\neg A \wedge B)), 3]$, which is logically equivalent to $[(\neg A), 3]$.

4.1 Lossy Operators

In order to fully exploit the power of structured representations, we need lossy operators. Note that without lossy operators, we cannot guarantee that the size of the computed message will be bounded by the size of the structure associated with each edge. Since lossy sum, product and division operators are simply lossy projections of their lossless counterparts, we only have to define the lossy projection operator.

Definition 3. Given a ADD (or sparse table) $\mathcal{R}$, let $[f_i, v_i]$ be the weighted feature associated with a leaf node (entry) i. Then, the **lossy projection** of a probability distribution ϕ on i is $v_i = \frac{1}{|\texttt{Sol}(f_i)|} \sum_{\mathbf{y} \in \texttt{Sol}(f_i)} \phi(\mathbf{y})$ where $\texttt{Sol}(f_i)$ is the set of assignments that are consistent with f_i. The **lossy** projection of ϕ on $\mathcal{R}$, denoted by $\mathcal{R}[\phi]$ is the lossy projection of ϕ on all leaf nodes (entries) of $\mathcal{R}$.

We can show that:

Theorem 1. *Given a ADD (or sparse table) $\mathcal{R}$, the lossy projection operator (see Definition 3) minimizes the KL divergence between ϕ and $\mathcal{R}[\phi]$. In other words, there exists no other ADD (or sparse table) that has the same structure as $\mathcal{R}$ but has smaller KL divergence.*

At first glance, the lossy projection operator may seem impractical because it involves computing the number of assignments that are consistent with a feature, i.e., it includes solving the $\#\mathcal{P}$-complete model counting problem. However, for sparse hash tables and ADDs, this is not an issue because model counting is constant time and linear time in the size of the representation respectively.

5 A STRUCTURED MESSAGE PASSING ALGORITHM

Clearly, in order to exploit the compactness of ADDs and sparse tables, a majority of the messages should contain determinism and/or CSI. To this end, we propose to artificially introduce CSI and determinism in the messages. Intuitively, if the introduced CSI and determinism captures most of the probability mass in the message, the resulting algorithm will be as accurate as cluster graph BP. However, its time and space complexity will be much smaller.

We cannot add zeros or determinism arbitrarily, however. Notice that in order to guarantee that the KL divergence between the exact message and all possible projected messages does not equal infinity, all configurations of variables that are consistent in the exact message should also be consistent in the projected message. Otherwise, the minimum KL distance will equal infinity. For example,

Example 1. Consider a cluster that is associated with an ADD representing two features $(A \vee B \vee C, 5)$ and $(\neg A \wedge \neg B \wedge \neg C, 0)$ and an edge with its neighbor that is associated with an ADD representing two features $(\neg A \wedge \neg B, v)$ and $(A \vee B, 0)$. No matter what value of v is selected, the KL divergence between the exact message obtained by summing out C from the cluster and any message projected on the structured representation will be infinity.

There are a number of ways in which we can add determinism so that all clusters and edges satisfy the consistency condition described above. We propose the following approach because of its simplicity: generate a set of samples and project them on each cluster and each edge. The projected samples define a constraint that all partial assignments that are not present in the generated samples have zero weight. For example,

Example 2. Consider a graphical model with three binary variables $\{X_1, X_2, X_3\}$. Let $\mathbf{S} = \{(0,1,1),(1,0,0),(1,0,1)\}$ be a set of samples over the three variables. Consider a cluster defined over two variables $\{X_1, X_2\}$. Then, the projection of $\mathbf{S}$ on the cluster is the relation: $\{(0,1),(1,0)\}$. The other two assignments $\{(1,1),(0,0)\}$ have zero weight. The ADD associated with this cluster will represent the following set of features: $\{(\neg X_1 \wedge X_2, v_1)$, $(X_2 \wedge \neg X_2, v_2)$, $((\neg X_1 \wedge \neg X_2) \vee (X_1 \wedge X_2), 0)\}$.

We can show that our approach that introduces determinism via sampling is correct and yields a structured EP algorithm. The only assumption we have to make is that each tuple having non-zero value in each function in the graphical model is included in the samples. This will ensure that the KL divergence between any function in the graphical model and its lossy projection is finite. Formally,

Theorem 2. *Given a graphical model $\mathcal{G} = (\mathbf{X}, \mathbf{D}, \Phi)$, a structured cluster graph $G(\mathbf{V}, \mathbf{E})$, let $\mathbf{S}$ be a set of samples over $\mathbf{X}$ such that: (1) For each cluster $V \in \mathbf{V}$ and each edge $E \in \mathbf{E}$, $\mathcal{R}_V$ and $\mathcal{R}_E$ are such that for any function ϕ and for all configurations $\mathbf{x} \notin \mathbf{S}$, $\mathcal{R}_V[\phi](\mathbf{x}) = 0$ and $\mathcal{R}_E[\phi](\mathbf{x}) = 0$ and for all configurations $\mathbf{x} \in \mathbf{S}$, $\mathcal{R}_V[\phi](\mathbf{x}) > 0$ and $\mathcal{R}_E[\phi](\mathbf{x}) > 0$ and (2) For each function $\phi \in \Phi$, all assignments $\mathbf{x} \in S(\phi)$ such that $\phi(\mathbf{x}) > 0$ are included in $\mathbf{S}$. Then, for each edge E, there exists a lossy message such that the KL divergence between the lossless message and the lossy one is finite.*

To artificially introduce CSI in the message, we propose to use quantization [10, 30]. In this approach, given a small real number ϵ, we put all values in the range of the function into multiple bins such that the absolute difference between any two values in each bin is bounded by ϵ. The goal is to minimize the number of bins. Then, we replace all values in each bin by their average value in each function. Quantization reduces the number of distinct values in the range of the function and as a result reduces the size of the ADD representing the message.

The discussion above yields Algorithm 1, which is a possible instance of SMP. The algorithm takes as input a graphical model $\mathcal{G} = (\mathbf{X}, \mathbf{D}, \Phi)$, a cluster graph $G(\mathbf{V}, \mathbf{E})$, a representation system $\mathcal{R}$, an integer k denoting the number of samples and a real number ϵ, which denotes the error bound used for quantization. The algorithm first generates samples from the graphical model. The samples can be generated using either importance sampling or Gibbs sampling. (For higher accuracy, the samples should be such that they

Algorithm 1: Structured Message Passing

```
Input: A graphical model $\mathcal{G} = (\mathbf{X}, \mathbf{D}, \Phi)$, a cluster graph
       $G = (\mathbf{V}, \mathbf{E})$, a representation system $\mathcal{R}$, integer $k > 0$
       and a real number $0 \leq \epsilon \leq 1$
Output: A structured cluster graph with (converged) messages
        and potentials
begin
    S=Generate $k$ Samples from $\mathcal{G}$;
    for each $V \in \mathbf{V}$ and $E \in \mathbf{E}$ do
        // Project $\mathbf{S}$ on $V$ and $E$
        Initialize $\mathcal{R}_V[\phi_V]$ and $\mathcal{R}_E[\phi_E]$ to zero;
        for all configurations $\mathbf{x} \in \mathbf{S}$ do
            set $\mathcal{R}_V[\phi_V](\mathbf{x}) = 1$ and $\mathcal{R}_E[\phi_E](\mathbf{x}) = 1$;
    Let $G' = (\mathbf{V}', \mathbf{E}')$ be the structured cluster graph obtained
    in the above step;
    for each function $\phi$ in $\Phi$ do
        Find a cluster $V \in \mathbf{V}'$ such that $S(\phi) \subseteq S(\phi_V)$ and
        multiply $\phi$ with $\mathcal{R}_V[\phi_V]$;
    Run sum-product or belief-update message passing on $G'$
    until convergence. Quantize each message using $\epsilon$;
```

capture the modes of the distribution.) After the samples are generated, we project the samples on each cluster and each edge and use the representation system to yield a structured cluster graph $G'(\mathbf{V}', \mathbf{E}')$. Then, we initialize the parameters of each cluster $V \in \mathbf{V}'$ and each edge $E \in \mathbf{E}'$ using the functions in the graphical model. Finally, the algorithm runs either sum-product or belief-update message passing on the structured cluster graph. Each message is quantized using ϵ.

Algorithm 1 is equivalent to the cluster graph BP algorithm when $k = \infty$ and $\epsilon = 0$. It is equivalent to the quantization-based EP algorithm proposed in [10] when $k = \infty$ and $\epsilon > 0$. For other values of k and ϵ, Algorithm 1 yields a Monte Carlo approximation of cluster graph BP and EP.

The algorithm just presented belongs to a class of algorithms that combine sampling-based inference with message-passing based inference. Many other advanced algorithms proposed in literature such as Particle BP [14, 31], AND/OR sampling [9], and sample propagation [23] belong to this class. The novelty in our proposed algorithm is that we combine sampling-based inference with message-passing inference over structured cluster-graphs. This yields several interesting bias versus variance tradeoffs, which can be leveraged to improve the accuracy of estimation. We discuss these tradeoffs next.

5.1 Analysis: Bias-Variance Tradeoffs

In this section, we analyze the bias-variance tradeoffs in Algorithm 1. Let $f(\mathbf{x})$ be the quantity that we want to estimate (e.g., the partition function or the posterior marginals). Given a set of samples $\mathbf{S}$, a cluster graph G and constant ϵ, let $h(\mathbf{x}; G, \epsilon, \mathbf{S})$ denote the estimate of $f(\mathbf{x})$ computed using Algorithm 1 with $G, \mathbf{S}, \epsilon$ as inputs. Then, the expected mean squared error between $f(\mathbf{x})$ and $h(\mathbf{x}; G, \epsilon, \mathbf{S})$ is

$$\mathbb{E}_{\mathbf{S}}[\{h(\mathbf{x}; G, \epsilon, \mathbf{S}) - f(\mathbf{x})\}^2] = \{\mathbb{E}_{\mathbf{S}}[h(\mathbf{x}; G, \epsilon, \mathbf{S})] - f(\mathbf{x})\}^2 + \{\mathbb{E}_{\mathbf{S}}[\{h(\mathbf{x}; G, \epsilon, \mathbf{S}) - \mathbb{E}_{\mathbf{S}}[h(\mathbf{x}; G, \epsilon, \mathbf{S})]\}^2]\} \quad (2)$$

The first term in Eq. (2) equals bias squared and the second term equals the variance.

We can show that:

Theorem 3. *Increasing the cluster size (or decreasing ϵ) of the cluster graph used by Algorithm 1 decreases the asymptotic bias $\lim_{|\mathbf{S}| \to \infty} |\mathbb{E}_{\mathbf{S}}[h(\mathbf{x}; G, \epsilon = 0, \mathbf{S})] - f(\mathbf{x})|$ but increases the variance.*

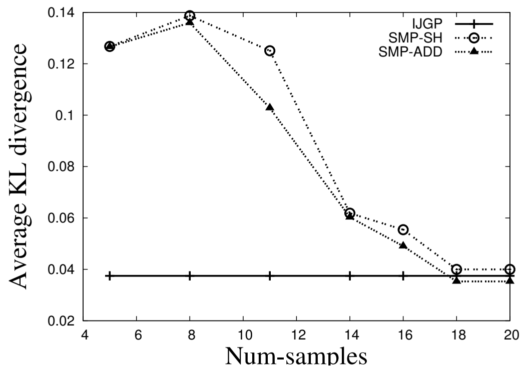
Proof. (Sketch) Notice that in the limit of infinite samples and assuming that $\epsilon = 0$, Algorithm 1 is equivalent to the cluster graph BP algorithm (we also assume that the sampling algorithm generates every assignment having non-zero probability in $\mathcal{G}$ with non-zero probability). Since the set of cluster graphs whose cluster size is bounded by i (along with the associated cluster potentials and messages) is included in the set of cluster graphs whose cluster size is bounded by $i+1$, the bias will never increase as we increase the cluster size. Moreover, cardinality arguments (the number of different clusters of size $i+1$ is far greater than the number of different clusters of size i) dictate that there exists a particular setting of cluster potentials and messages in a cluster graph whose cluster size is bounded by $i+1$ that cannot be represented by any cluster graph whose cluster size is bounded by i and therefore the asymptotic bias decreases as we increase the cluster size. (The asymptotic bias of a junction tree is zero).

To prove that the variance increases as we increase the cluster size, consider two clusters V and V' where V' is constructed from V by adding a variable to it. Clearly, projecting the given set $\mathbf{S}$ of samples on V will cover a greater percentage of the tuples in the potential associated with V as compared to the potential associated with V'. As a result, the effective sample size at V is larger than the effective sample size at V'. Since the variance decreases as the sample size increases, the variance at V will be smaller than the variance at V'. Therefore, the variance increases as we increase the cluster size. Increasing ϵ has the effect of introducing new context specific (conditional) independences, which is the same as decreasing the cluster size. Therefore, the same arguments apply to decreasing ϵ. □

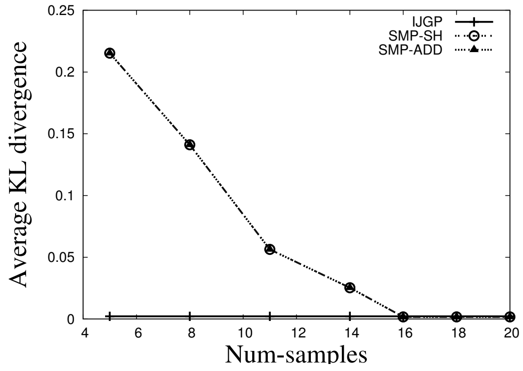
From the central limit theorem, it is immediate that:

Theorem 4. *Increasing the number of samples while fixing G and ϵ decreases the variance. Moreover, the sample bias converges to the asymptotic bias at the rate of $O(|\mathbf{S}|^{-1/2})$.*

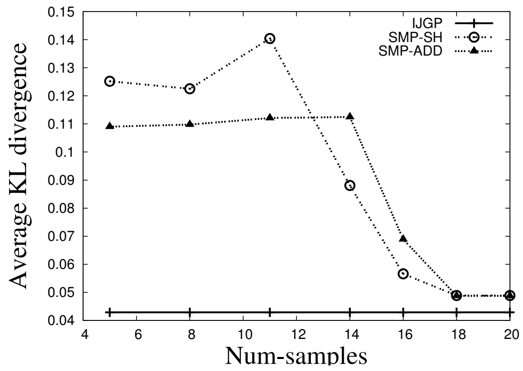
Theorems 3 and 4 summarize the bias-variance tradeoffs associated with Algorithm 1. For a fixed sample size, cluster graphs having large clusters will typically have low bias and high variance while cluster graphs having small clusters will typically have large bias and low variance.



(a)

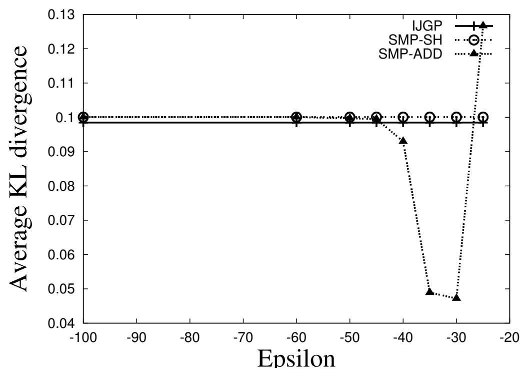


(b)

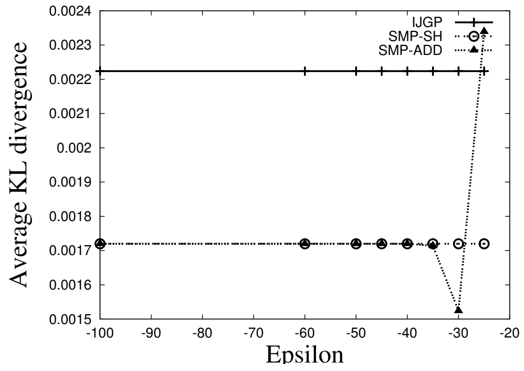


(c)

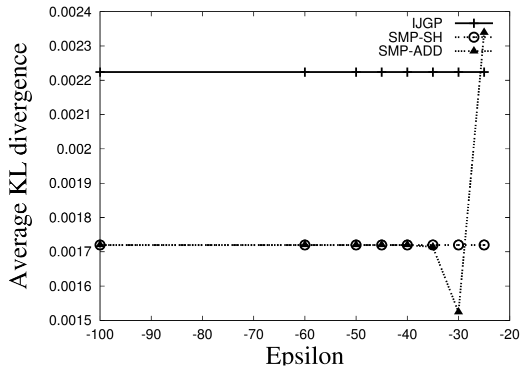
Figure 2: Impact of varying k (number of samples), keeping the i-bound and ϵ constant for (a) 20×20 Ising model with 10 evidence nodes, $i = 9$ and $\epsilon = 2^{-40}$, (b) Block coding instance with 255 variables and 511 functions, $i = 12$ and $\epsilon = 2^{-100}$ and (c) Logistics instance with 1413 nodes and 29487 functions, $i = 15$ and $\epsilon = 2^{-40}$.



(a)



(b)

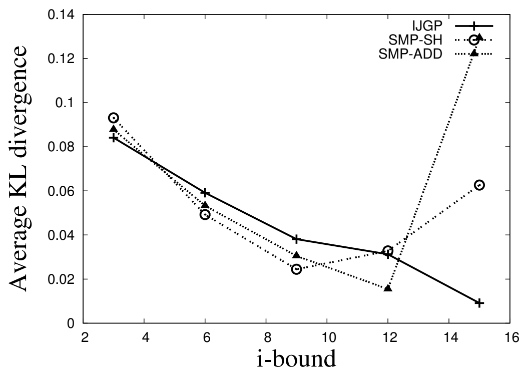


(c)

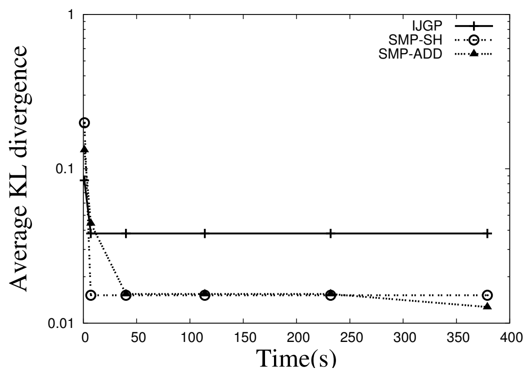
Figure 3: Impact of varying the quantization parameter ϵ, keeping the i-bound and k constant for (a) 20×20 Ising model with 10 evidence nodes, $i = 6$ and $k = 2^5$, (b) Block coding instance with 255 variables and 511 functions, $i = 12$ and $k = 2^{16}$ and (c) Logistics instance with 1413 nodes and 29487 functions, $i = 12$ and $k = 2^{16}$.

6 EXPERIMENTS

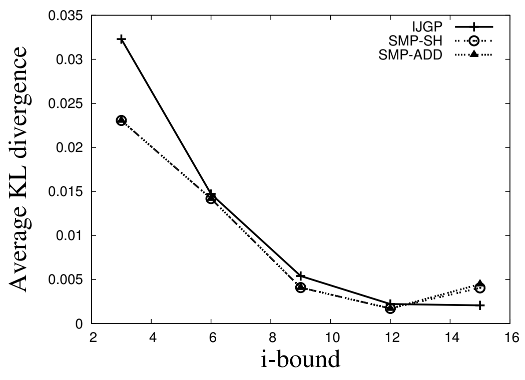
In this section, we compare SMP with Iterative Join Graph propagation (IJGP) [19], a state-of-the-art tabular cluster graph BP algorithm. IJGP won two out of the three marginal estimation categories in the 2010 UAI competition [8]. We experimented with instances from three benchmark domains: (i) Ising models (these instances are available from the PASCAL 2011 probabilistic inference challenge), (ii) linear block coding (these instances available from the UAI 2008 evaluation), and (iii) logistic planning (these instances are available from the authors of Cachet [25]). Ising models have no determinism, the linear



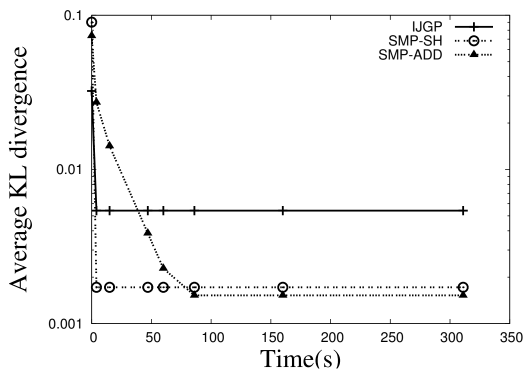
(a)



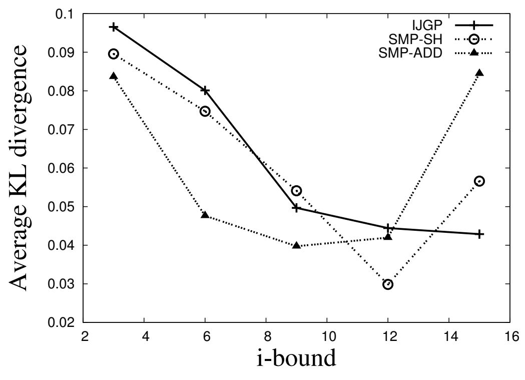
(a)



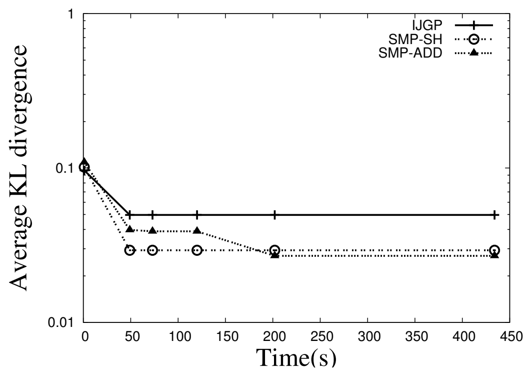
(b)



(b)



(c)



(c)

Figure 4: Impact of varying the i-bound, keeping k and ϵ constant for (a) 20×20 Ising model with 5 evidence nodes, $k = 2^{14}$ and $\epsilon = 2^{-30}$, (b) Block coding instance with 255 variables and 511 functions, $k = 2^{18}$ and $\epsilon = 2^{-35}$ and (c) Logistics instance with 1413 nodes and 29487 functions, $k = 2^{16}$ and $\epsilon = 2^{-35}$.

Figure 5: KL-divergence as a function of time (a) 21×21 Ising model with 5 evidence nodes, (b) Block coding instance with 255 variables and 511 functions and (c) Logistics instance with 1413 nodes and 29487 functions.

block coding networks have determinism and CSI while the logistic planning instances have determinism but no CSI.

We used the CUDD package [29] to implement ADDs. For a fair comparison, we constructed the cluster graphs for SMP using the same approach used by IJGP (see [19] for details). In IJGP, the complexity of inference is controlled by bounding the number of variables in each cluster by an integer parameter i, called its i-bound. We varied the i-bound from 3 to 15 in increments of 3. For the SMP algorithm, we varied ϵ (which controls quantization) from 2^{-20} to 2^{-100} and k (number of samples) from 2^5 to 2^{20}. We used Gibbs sampling on the Ising models and importance sampling on the other two for generating samples (This is because Gibbs sampling does not converge in presence of determinism). We ran each algorithm until convergence or until 15 minutes or until it exceeded a memory bound of 512 MB, whichever was earlier. We chose these values because the benchmarks can be solved exactly in roughly 1 hour of cpu time using up to 8 GB of RAM. We used average KL divergence between the exact and approximate marginal distribution to measure accuracy.

For lack of space, we only show a fraction of our results in Figures 2-5. SMP-SH and SMP-ADD denote the sparse hash table based and ADD based implementation of SMP respectively. Note that each point in each figure denotes an average over 10 random runs of IJGP, SMP-SH and SMP-ADD respectively.

Figure 2 shows the impact of varying the number of samples k, keeping the i-bound and ϵ constant for three instances, one from each domain. As expected, the accuracy of SMP-SH and SMP-ADD improves with more samples.

Figure 3 shows the impact of varying the quantization parameter ϵ, keeping the i-bound and k constant for three instances. In all three cases, we clearly see the bias versus variance trade-off; as we decrease ϵ, the accuracy first improves and then reduces before stabilizing to a fixed point.

Figure 4 shows the impact of varying the i-bound, keeping ϵ and k constant for three instances. Again in all three cases, we clearly see the bias versus variance trade-off; as we increase the cluster size (i-bound), the accuracy improves until a certain i-bound after which it starts decreasing.

Figure 5 shows the accuracy of various schemes as a function of time. For each time-point, we select the parameters that yield the best accuracy for each of the three methods. SMP-ADD is more accurate than SMP-SH which in turn is more accurate than IJGP. Note the log-scale on the Y-axis and therefore there is an order of magnitude difference.

7 RELATED WORK

Our work is related to the work on structured region graphs (SRGs) by Welling et al. [33]. In it, the authors showed that Yedidia et al.'s [34] generalized belief propagation algorithm *morphs* into the EP algorithm [20] if each message and cluster potential in the region graph is approximated by a product of tractable tabular functions. Our work is different from Welling et al.'s work in that we propose to use structured representations which are often more compact than the product of tabular functions representation. Also, unlike our formulation, there is no straight-forward way of introducing and exploiting determinism and CSI in the SRG formalism.

Our work is also related to the work on non-parametric BP by Sudderth et al. [31] and particle BP by Ihler and McAllester [14], in which the authors propose to represent BP messages using samples or particles. However, unlike our work, these approaches do not exploit structured representations and do not utilize both CSI and determinism. Also, they do not investigate bias versus variance trade-offs as we do. Our work connects particle BP with EP, yielding a more unified perspective.

Another related work is that of [7, 12, 23] who perform sampling based inference on junction trees. The main idea in these papers is to perform message passing on a junction tree by substituting messages which are too hard to compute exactly by their sampling-based approximations. Unlike our work, however, they do not perform message passing over arbitrary cluster graphs. This is problematic because as we showed, for a small sample size, junction trees will have low bias but high variance and as a result they will likely yield inaccurate results.

Finally, our work is related to the work on approximation by quantization (ABQ) by Gogate and Domingos [10]. Unlike ABQ which only introduces CSI, we propose to introduce both CSI and determinism which as we show often yields better accuracy in practice.

8 SUMMARY AND FUTURE WORK

In this paper, we proposed *structured message passing*, a unifying approach for taking advantage of structured representations. We investigated the use of two structured representations within this framework: algebraic decision diagrams (ADDs) and sparse hash tables. ADDs are useful in the presence of CSI and/or determinism while sparse tables are useful only in the presence of determinism. Therefore, in order to fully utilize the power of ADDs and sparse tables, we proposed a new algorithm that artificially introduces CSI via quantization and determinism via sampling. Our new algorithm is quite powerful and includes the cluster graph BP algorithm, the EP algorithm and the particle BP algorithm as special cases. Our algorithm introduces several bias versus variance tradeoffs. We investigated these tradeoffs both theoretically and empirically and showed that our new algorithm is superior to state-of-the-art approaches such as Iterative Join Graph Propagation [19].

Future work includes: applying our algorithm to continuous and hybrid discrete/continuous graphical models; using other structured representations such as mixture models and Affine ADDs [26] within SMP; combining SMP with lifted inference (cf. [11]); using our algorithm for weight learning; developing automatic tuning strategies for finding the right balance between bias and variance; etc.

Acknowledgements

This research was partly funded by ARO grant W911NF-08-1-0242, AFRL contracts FA8750-09-C-0181 and FA8750-13-2-0019, NSF grant IIS-0803481, and ONR grant N00014-12-1-0312. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, AFRL, NSF, ONR, or the United States Government.

References

[1] R. Bahar, E. Frohm, C. Gaona, G. Hachtel, E. Macii, A. Pardo, and F. Somenzi. Algebraic decision diagrams and their applications. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 188–191, 1993.

[2] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-Specific Independence in Bayesian Networks. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, pages 115–123, 1996.

[3] M. Chavira and A. Darwiche. Compiling Bayesian Networks Using Variable Elimination. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 2443–2449, 2007.

[4] M. Chavira and A. Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6-7):772–799, 2008.

[5] A. Culotta, A. McCallum, B. Selman, and A. Sabharwal. Sparse message passing algorithms for weighted maximum satisfiability. In *New England Student Colloquium on Artificial Intelligence (NESCAI)*, 2007.

[6] A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

[7] A. P. Dawid, U. Kjaerulff, and S. L. Lauritzen. Hybrid Propagation in Junction Trees. In *Advances in Intelligent Computing*, pages 85–97, 1994.

[8] G. Elidan and A. Globerson. The 2010 UAI Approximate Inference Challenge. Available online at: http://www.cs.huji.ac.il/project/UAI10/index.php, 2010.

[9] V. Gogate and R. Dechter. AND/OR Importance Sampling. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 212–219, 2008.

[10] V. Gogate and P. Domingos. Approximation by Quantization. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 247–255, 2011.

[11] V. Gogate and P. Domingos. Probabilistic Theorem Proving. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 256–265. AUAI Press, 2011.

[12] L. D. Hernandez and S. Moral. Mixing Exact and Importance Sampling Propagation Algorithms in Dependence Graphs. *International Journal of Approximate Reasoning*, 12(8):553–576, 1995.

[13] C. Huang and A. Darwiche. Inference in Belief Networks: A Procedural Guide. *International Journal of Approximate Reasoning*, 15(3):225–263, 1996.

[14] A. T. Ihler and D. A. McAllester. Particle Belief Propagation. *Twelfth International Conference on Artificial Intelligence and Statistics*, pages 256–263, 2009.

[15] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[16] D. Larkin and R. Dechter. Bayesian inference in presence of determinism. In *Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[17] S. L. Lauritzen and D. J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 50(2):157–224, 1988.

[18] D. Lowd and P. Domingos. Approximate inference by compilation to arithmetic circuits. In *Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1477–1485, 2010.

[19] R. Mateescu, K. Kask, V. Gogate, and R. Dechter. Iterative Join Graph Propagation algorithms. *Journal of Artificial Intelligence Research*, 37:279–328, 2010.

[20] T. P. Minka. Expectation Propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 362–369, 2001.

[21] T. P. Minka and Y. Qi. Tree-structured Approximations by Expectation Propagation. In *Proceedings of the 17th Annual Conference on Neural Information Processing Systems (NIPS)*, 2003.

[22] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy Belief Propagation for Approximate Inference: An Empirical Study. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 467–475, 1999.

[23] M. A. Paskin. Sample Propagation. In *Advances in Neural Information Processing Systems*, pages 425–432, 2003.

[24] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA, 1988.

[25] T. Sang, P. Beame, and H. Kautz. Solving Bayesian networks by weighted model counting. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, pages 475–482, 2005.

[26] S. Sanner and D. A. McAllester. Affine algebraic decision diagrams (aadds) and their application to structured probabilistic inference. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 1384–1390, 2005.

[27] S. Sanner, W. T. B. Uther, and K. V. Delgado. Approximate dynamic programming with affine ADDs. In *Nineth International Conference on Autonomous Agents and Multiagent Systems*, pages 1349–1356, 2010.

[28] A. Silberschatz, H. F. Korth, and S. Sudarshan. *Database System Concepts, 4th Edition*. McGraw-Hill Book Company, 2001.

[29] F. Somenzi. CUDD: CU Decision Diagram Package Release 2.2.0, 1998.

[30] R. St-aubin, J. Hoey, and C. Boutilier. APRICODD: Approximate policy construction using decision diagrams. In *In Proceedings of Conference on Neural Information Processing Systems*, pages 1089–1095, 2000.

[31] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky. Nonparametric Belief Propagation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 605–612, 2003.

[32] T. Walsh. SAT v CSP. In *Sixth International Conference on Principles and Practice of Constraint Programming*, pages 441–456, 2000.

[33] M. Welling, T. P. Minka, and Y. W. Teh. Structured Region Graphs: Morphing EP into GBP. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, pages 609–614, 2005.

[34] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized Belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, 2005.

Multiple Instance Learning by Discriminative Training of Markov Networks

Hossein Hajimirsadeghi, Jinling Li, Greg Mori
School of Computing Science
Simon Fraser University

Mohamed Zaki, Tarek Sayed
Department of Civil Engineering
University of British Columbia

Abstract

We introduce a graphical framework for multiple instance learning (MIL) based on Markov networks. This framework can be used to model the traditional MIL definition as well as more general MIL definitions. Different levels of ambiguity – the portion of positive instances in a bag – can be explored in weakly supervised data. To train these models, we propose a discriminative max-margin learning algorithm leveraging efficient inference for cardinality-based cliques. The efficacy of the proposed framework is evaluated on a variety of data sets. Experimental results verify that encoding or learning the degree of ambiguity can improve classification performance.

1 INTRODUCTION

Multiple instance learning aims to recognize patterns from weakly supervised data. Contrary to standard supervised learning, where each training instance is labeled, in the MIL paradigm training instances are given in positive and negative bags. In the traditional MIL definition, a bag is positive if it contains at least one positive instance, while in a negative bag all the instances are negative. This ambiguity in the instance labels is passed on to the learning algorithm, which should incorporate the information to classify unseen bags. In this paper we develop a novel framework for MIL with a more general definition of a positive bag.

Multiple instance learning has been successfully used in many applications such as image categorization (Chen et al., 2006), text categorization (Andrews et al., 2002), content-based image retrieval (Zhang and Goldman, 2002), text-based image retrieval (Li et al., 2011; Duan et al., 2011), object detection (Viola et al., 2006) and tracking (Babenko et al., 2009). Chen et al. (2006) treated each image as a bag of instances corresponding to blocks, regions, or patches of the image for the purpose of image categorization. Andrews et al. (2002) approached text categorization with a MIL framework, where each document is represented by a bag of passages. Li et al. (2011) and Duan et al. (2011) used MIL to handle ambiguity in labels of training images incurred by coarse ranking of web images. Viola et al. (2006) used MIL to overcome the ambiguity in object annotation, by representing each image with a bag of windows centered around the ground truth. Likewise, in object tracking Babenko et al. (2009) used several blocks around the estimated object location to construct a positive training bag for MIL.

The traditional MIL definition states that *at least one* of the instances in a positive bag is positive. However, this is a too weak statement in many MIL applications. For example, in image retrieval most top-ranked training images are truly relevant to the query – they are true positives and not just additional irrelevant elements in a bag (Li et al., 2011). Using this prior information can help to train stronger and more robust classifiers. Further, in some applications, because of noisy, imperfect, or low-quality feature representations, negative bags can contain instances that are effectively indistinguishable from positive instances. In these situations more robust MIL definitions are needed.

To address these issues, we develop a MIL framework based on Markov networks with a flexible notion of a positive bag. This general MIL framework uses cardinality-based measurements over bags, which extend from the notion of “at least one positive” to “at least some positives” to “nearly all positives.” Thus, it can explore different levels of ambiguity in the data. In addition, this framework can be adapted to estimate the appropriate MIL notion from training data without prior information about the fraction of positives in the bags. We show that it is possible to use efficient inference techniques (Gupta et al., 2007) to train and evaluate these general MIL models quickly. For the learning criterion, we propose a max-margin discriminative algorithm to train the models.

This paper is organized as follows. Section 2 reviews related work. Section 3 describes our framework of multiple instance learning with Markov networks. In particular, the models for different MIL definitions, including the traditional MIL definition and more general MIL definitions are described in this section. In Section 4 the inference and learning algorithms are explained. Section 5 provides experimental comparisons to state-of-the-art MIL algorithms and an application to video sequence classification. We conclude in Section 6.

2 RELATED WORK

Dietterich et al. (1997) introduced the first algorithms for multiple instance learning. The main idea was to construct a hyper-rectangle maximizing the number of enclosed instances from positive bags while excluding all the instances of negative bags. Based on similar ideas, the general diverse density (DD) framework (Maron and Lozano-Pérez, 1998) was proposed. This algorithm works by finding a concept point which is near to at least one instance of every positive bag, but far from all negative instances. Next, EM-DD (Zhang and Goldman, 2002), the expectation-maximization version of DD, was proposed by incorporating the iterative EM approach of estimating positive instances and refining the concept hypothesis within the DD framework.

Gärtner et al. (2002) defined a kernel for multiple instance data and used SVMs to learn a bag classifier. Andrews et al. (2002) modified SVMs for MIL, proposing two algorithms. The first, mi-SVM, aims to maximize the instance margin jointly over the hidden instance labels. The second, MI-SVM, tries to maximize the bag margin, where the bag margin is defined by the most positive instance of each bag. Chen et al. (2006) employed a DD function to map the instances of a bag into a bag-level feature vector. Then, the important features were used by 1-norm SVM for image categorization. Zhou et al. (2009) proposed MIGraph and miGraph. In these methods, first a graph is constructed for each bag, and then an SVM is trained by designing a graph kernel. Thus, by considering the relations among the instances in a bag, the instances are treated as non-i.i.d samples.

The very successful Latent SVM (Felzenszwalb et al., 2010) is also a multiple instance learning framework. For positive instances, a set of latent variable values is used. One can consider the set of completed data instances (latent variable values with observed input feature values) as a "bag" in MIL, as in the MI-SVM framework (Andrews et al., 2002). Latent SVMs have been used in numerous applications, and often obtain state of the art performance. However, they use the "at least one positive instance" positive bag definition. As noted above, for some applications this is limiting since many latent variable settings could in fact be positive and could aid in training a better classifier. The more general MIL definition and algorithms in this paper aim to remedy this.

In recent years, more advanced algorithms have been developed to address non-traditional MIL definitions. Gehler and Chapelle (2007) proposed SVM-like algorithms, AL-SVM and AW-SVM, for MIL. They argued that different levels of ambiguity in positive bags can influence the performance of MIL-based methods. Hence, they provided the possibility to encode prior knowledge about the data set, i.e., fraction of positive instances (witnesses) in a bag. However, these algorithms need a preset parameter which determines the fixed ratio of witnesses.

Bunescu and Mooney (2007) used the transductive SVM framework to propose a MIL algorithm for sparse positive bags. Li and Sminchisescu (2010) proposed a MIL model based on likelihood ratio estimation. The likelihood ratio is estimated by a support vector regression scheme. For bag classification, an SVM is trained to linearly combine the instance likelihood ratios in a postprocessing step. Although, the original model formulation follows the traditional MIL assumption, however, their experiments show that the postprocessing makes this algorithm suboptimally adaptive to different witness rates.

Duan et al. (2011) and Li et al. (2011) introduced a generalized MIL definition, where the positive bags contained at least a certain portion of positive instances. They used a mixed-integer SVM formulation with new constraints on instance labels of the bags. It is shown that this NP-hard problem can be viewed as a multiple kernel learning problem with an exponential number of base kernels. Thus, this algorithm requires some heuristics to solve the original problem. Hajimirsadeghi and Mori (2012) proposed a boosting algorithm for MIL, which can softly explore different levels of ambiguity using linguistic aggregation functions with different degrees of orness. However, this algorithm also needs approximate before-hand knowledge of the witness ratio, or uses cross-validation to estimate it.

In this work, we propose a MIL framework based on Markov networks. This framework is used to model more general MIL definitions, and superior to previous algorithms (Gehler and Chapelle, 2007; Duan et al., 2011; Li et al., 2011; Hajimirsadeghi and Mori, 2012), it can also work without prior information about the fraction of positive instances inside the bags. In fact, the proposed model can be trained to discover this knowledge directly from data. The inference and learning of the proposed models is exact and no heuristics are needed. Further, the Markov network allows flexible modification and extensions, for instance modeling bag structure. This framework could also be modified to address issues such as training individual classifiers from group statistics of label proportions (Kueck and de Freitas, 2005; Quadrianto et al., 2009; Rueping, 2010).

Note that there are also some other MIL methods based on Markov networks or conditional random fields (CRFs). Deselaers and Ferrari (2010) proposed MI-CRF. In this method, the bags are modelled as nodes in a CRF, where each node can take one of the instances in the bag as its state. So, the bags are jointly trained and classified in this model. Warrell and Torr (2011) proposed another CRF-based method. This method provides a structured bag model, by constructing an undirecetd graph among the instances, instance labels and the bag label. In this CRF, hard and soft MIL constraints are incorporated in the model by defining energy functions between the labels. However, infering the proposed CRFs is performed *approximately* by dual decomposition, and the models are trained by deterministic annealing. Tarlow et al. (2012) proposed a model to approach MIL by CRFs with cardinality potentials over instance labels. However, this model works by sum-product (i.e., marginalization) inference of cardinality potentials. Note that maximum a posteriori (MAP) inference of cardinality potentials, which is used in our proposed method, is faster than the sum-product inference (Tarlow et al., 2012). In addition, our max-margin learning algorithm is different from their maximum likelihood approach to learning.

3 MIL USING MARKOV NETWORKS

In MIL, training examples are presented in bags where the instances in a bag share a label. In this work, we use Markov networks to model MIL problems and develop a generalized notion of positive bags.

The Markov network is used to define a scoring function for a bag. A graphical representation of the proposed Markov network for a bag is shown in Figure 1. Each instance and its label are modeled by two nodes in a clique. The clique potential specifies a classifier for an individual instance. A second clique contains all instance labels and the bag label. This clique is used to define what makes a bag positive. Varying this clique potential will lead to different MIL definitions, and is the focus for our work.

3.1 MODEL DETAILS

More formally, let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$ denote a bag with m instances and a binary bag label $y \in \{-1, 1\}$. The collective binary instance labels are denoted by $\mathbf{h} = \{h_1, \cdots, h_m\}$. We use the Markov network in Figure 1 to define a scoring function over tuples $(\mathbf{X}, \mathbf{h}, y)$. In testing, this scoring function will be used to find the label y for a test bag, inferring the bag and instance labels that maximize the scoring function.

The network has cliques on each instance and its label, and one clique on all instance labels and the bag label. We define the scoring function on these cliques by:

$$f_{\mathbf{w}}(\mathbf{X}, \mathbf{h}, y) = \phi^C_{\mathbf{w}}(\mathbf{h}, y) + \sum_i \phi^I_{\mathbf{w}}(\mathbf{x}_i, h_i), \qquad (1)$$

where $\phi^I_{\mathbf{w}}(\mathbf{x}_i, h_i)$ represents the potential between each instance and its label, and $\phi^C_{\mathbf{w}}(\mathbf{h}, y)$ is the clique potential over all the instance labels and the bag label. Note that the potential functions in (1) are parametrized by $\mathbf{w}$. We explain the details of these potential functions as follows.

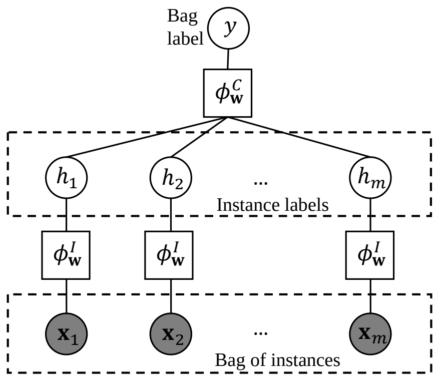


Figure 1: Graphical illustration of the proposed model for multiple instance learning. Potential functions relate instances $\mathbf{x}_i$ to labels h_i. A clique relates all instance labels h_i to the bag label y.

Instance-Label Potential $\phi^I_{\mathbf{w}}(\mathbf{x}_i, h_i)$: This potential function models the compatibility between the ith instance feature vector $\mathbf{x}_i$ and its label h_i. It is parametrized as:

$$\begin{aligned} \phi^I_{\mathbf{w}}(\mathbf{x}_i, h_i) &= \mathbf{w}_I^\top \mathbf{x}_i \, h_i \\ &= \mathbf{w}_I^\top \psi_I(\mathbf{x}_i, h_i). \end{aligned}$$

Labels Clique Potential $\phi^C_{\mathbf{w}}(\mathbf{h}, y)$: This potential function models the relations between the instance labels and the bag label. Since the MIL problems are defined based on the number of positive and negative instances, we can formulate this as a *cardinality-based* clique potential. Cardinality-based potentials are only a function of counts – in this case, the counts of the numbers of positive and negative instances in the bag.

By modifying the form of the cardinality-based potential, we can obtain different MIL definitions, which will be shown in the subsequent section. Moreover, while for arbitrary clique potentials inference could be NP-complete, for cardinality-based potentials efficient inference algorithms exist. This will lead to efficient algorithms for training and testing, described in Section 4.

In order to define the cardinality-based potentials, we will use the notation m^+/m^- for the number of labels in $\mathbf{h}$ which are positive/negative. The clique potential depends on these counts, and the bag label y. We parameterize two different clique potentials, one for positive bags ($C^+_{\mathbf{w}}$) and one for negative bags ($C^-_{\mathbf{w}}$):

$$\begin{aligned}\phi^C_{\mathbf{w}}(\mathbf{h}, y) &= C_{\mathbf{w}}\left(m^+, m^-, y\right)\\ &= C^+_{\mathbf{w}}\left(m^+, m^-\right)\mathbb{1}(y=1) \\ &+ C^-_{\mathbf{w}}\left(m^+, m^-\right)\mathbb{1}(y=-1).\end{aligned} \tag{2}$$

The following sections define functions $C^+_{\mathbf{w}}$ and $C^-_{\mathbf{w}}$ that lead to a variety of MIL models.

3.1.1 Multiple Instance Markov Network (MIMN)

This network models the standard MIL problem, i.e., in a positive bag at least one of the instances is positive, and in a negative bag all the instances are negative. The labels clique potential is given by

$$C^+_{\mathbf{w}}(0, m) = -\infty \tag{3}$$
$$C^+_{\mathbf{w}}(m^+, m - m^+) = w^+_c \quad m^+ = 1, \cdots, m \tag{4}$$
$$C^-_{\mathbf{w}}(0, m) = w^-_c \tag{5}$$
$$C^-_{\mathbf{w}}(m^+, m - m^+) = -\infty \quad m^+ = 1, \cdots, m. \tag{6}$$

This clique potential states that in a positive bag it is impossible to have all the instances be negative (3), but there is the same potential of having more than one positive instance (4). However, for a negative bag, it is only possible to have negative instances (5) & (6). One might set w^+_c and w^-_c to a constant value (e.g. 0), but we treat them as the model parameters and show how to learn them in Section 4.2.

3.1.2 Ratio-constrained Multiple Instance Markov Network (RMIMN)

Ratio-constrained MIL extends the notion of positive bags in MIL. In RMIMN, each positive bag contains at least a certain portion of positive instances. For example, at least $x\%$ of the instances should be positive in a positive bag. To model this problem with our proposed Markov network, we can refine the functions C^+ and C^-:

$$\begin{aligned} C^+_{\mathbf{w}}(m^+, m - m^+) &= -\infty \quad 0 \le \frac{m^+}{m} < \rho \\ C^+_{\mathbf{w}}(m^+, m - m^+) &= w^+_c \quad \rho \le \frac{m^+}{m} \le 1 \\ C^-_{\mathbf{w}}(m^+, m - m^+) &= w^-_c \quad 0 \le \frac{m^+}{m} < \rho \\ C^-_{\mathbf{w}}(m^+, m - m^+) &= -\infty \quad \rho \le \frac{m^+}{m} \le 1, \end{aligned} \tag{7}$$

where ρ indicates the required portion of positive instances in a positive bag.

3.1.3 Generalized Multiple Instance Markov Network (GMIMN)

GMIMN allows a very flexible notion of positive bags. We allow the portions of positive and negative instances in bags to be a learned parameter, discovered from the data. The MIL model will learn which fractions of instances tend to be positive in a bag. This network provides a very general model for multiple instance learning and is parametrized by:

$$\begin{aligned} &C^+_{\mathbf{w}}(0, m) = -\infty \\ &C^+_{\mathbf{w}}(m^+, m - m^+) = \sum_{k=1}^{K} w^+_k \mathbb{1}\left(\frac{k-1}{K} < \frac{m^+}{m} \le \frac{k}{K}\right) \\ &\qquad\qquad m^+ = 1, \cdots, m \\ &C^-_{\mathbf{w}}(m^+, m - m^+) = \sum_{k=1}^{K} w^-_k \mathbb{1}\left(\frac{k-1}{K} \le \frac{m^+}{m} < \frac{k}{K}\right) \\ &\qquad\qquad m^+ = 0, \cdots, m-1 \\ &C^-_{\mathbf{w}}(m, 0) = -\infty. \end{aligned} \tag{8}$$

where K determines the number of weighted segments of a bag. This model divides the bag size into K equal parts, and the weight of each segment w_k determines how important it is that the number of positive instances be placed inside that interval[1]. In other words,

[1] Note that the weights w_k are not necessarily monotonically increasing as k increases. For example, in a MIL data set, there might be only a few true positive instances in the positive bags, and so the potential of having many instances be positive is low.

these learning weights specify the importance or impact of different witness ratios for labeling a bag as positive or negative. Large values of K provide more detailed and specific models of bag definition by learning cardinality-based measures with finer resolution, while low values of K define a coarser model of bag. So, by controlling the granularity, this parameter is set in a trade-off between training accuracy and generalization ability[2]. Note that $C^{+}_{\mathbf{w}}(0, m) = -\infty$ and $C^{-}_{\mathbf{w}}(m, 0) = -\infty$ are the only required prior information in this model.

With these definitions, we note that using $C^{+}_{\mathbf{w}}$ and $C^{-}_{\mathbf{w}}$ defined in any of the MIMN, RMIMN, or GMIMN models makes the clique potential (i.e., $\phi^{C}_{\mathbf{w}}$) a linear function of the learning parameters. More formally:

$$\phi^{C}_{\mathbf{w}}(\mathbf{h}, y) = \mathbf{w}_C^{\top} \Psi_C(\mathbf{h}, y) + g_C(\mathbf{h}, y), \qquad (9)$$

where $\mathbf{w}_C$ represents the concatenation of the learning parameters in $C^{+}_{\mathbf{w}}$ and $C^{-}_{\mathbf{w}}$, while $\Psi_C(\mathbf{h}, y)$ and $g_C(\mathbf{h}, y)$ are functions independent of $\mathbf{w}$, which are specified by aggregation of the indicator functions.

4 INFERENCE AND LEARNING

The MIL models above define scoring functions that consider counts of instance labels in a bag. Using this, for a given bag we can define a scoring function for labeling a bag $\mathbf{X}$ with a label y:

$$F_{\mathbf{w}}(\mathbf{X}, y) = \max_{\mathbf{h}} f_{\mathbf{w}}(\mathbf{X}, \mathbf{h}, y). \qquad (10)$$

Below, we describe how to use efficient inference algorithms (Gupta et al., 2007) to efficiently solve this inference problem for the cardinality-based cliques we defined above.

Using this inference technique, learning can be performed using a max-margin criterion, as in the Latent SVM approach.

Classification of a new test bag can be done in a similar manner. We can predict the bag label by simply running inference, trying $y = +1$ and $y = -1$ and taking the maximum scoring bag label:

$$y^{\star} = \arg\max_{y} F_{\mathbf{w}}(\mathbf{X}, y). \qquad (11)$$

4.1 INFERENCE

The inference problem is to find the best set of instance labels $\mathbf{h}$ given observed values for the data instances $\mathbf{X}$ and the bag label y – the maximization problem in (10). Using (1) and (2), the inference problem can be written as

$$\max_{\mathbf{h}} \sum_i \phi^{I}_{\mathbf{w}}(\mathbf{x}_i, h_i) + C_{\mathbf{w}}(m^{+}, m^{-}, y). \qquad (12)$$

This problem is an instance of inference in graphical models with cardinality-based clique potentials (Gupta et al., 2007). This class of clique potentials is specified by two parts: the sum of individual node potentials and a function over all the nodes which only depends on the counts of the nodes which get specific labels. Efficient inference algorithms have been proposed for this class of graphical model in (Gupta et al., 2007). In this paper, we only work with the binary case (i.e., $h_i \in \{+1, -1\}$), for which there is an exact inference algorithm with $O(m \log m)$ time complexity. The inference algorithm is as follows.

First, sort the instances in decreasing order of $\phi^{I}_{\mathbf{w}}(\mathbf{x}_i, +1) - \phi^{I}_{\mathbf{w}}(\mathbf{x}_i, -1)$. Then, for $k = 0, \cdots, m$, compute s_k, the sum of the top-k instance potentials $\phi^{I}_{\mathbf{w}}(\mathbf{x}_i, +1) - \phi^{I}_{\mathbf{w}}(\mathbf{x}_i, -1)$ plus the clique potential $C_{\mathbf{w}}(k, m-k, y)$. Finally, find $k^{\star}$ which gets the largest s_k, and inference is accomplished by assigning the top $k^{\star}$ instances to positive labels and the rest to negative labels.

4.2 LEARNING

The training set is given by $\{(\mathbf{X}^1, y^1), \cdots, (\mathbf{X}^N, y^N)\}$, and the goal is to train the Markov models by learning the parameters $\mathbf{w}$. Inspired by the relations to latent SVM, we formulate the learning problem as minimizing the regularized hinge loss function:

$$\begin{aligned} &\min_{\mathbf{w}} \sum_{n=1}^{N} (\mathcal{L}^n - \mathcal{R}^n) + \frac{\lambda}{2}\|\mathbf{w}\|^2 \\ &\text{where } \mathcal{L}^n = \max_{y} \max_{\mathbf{h}} (\Delta(y, y^n) + f_{\mathbf{w}}(\mathbf{X}^n, \mathbf{h}, y)), \\ &\mathcal{R}^n = \max_{\mathbf{h}} f_{\mathbf{w}}(\mathbf{X}^n, \mathbf{h}, y^n), \\ &\Delta(y, y^n) = \begin{cases} 1 & \text{if } y \neq y^n \\ 0 & \text{if } y = y^n. \end{cases} \end{aligned} \qquad (13)$$

One approach to solve this problem approximately is the iterative algorithm of alternating between inference of the latent variables and optimization of the model parameters. So, the first step estimates the instance labels and the second step learns a standard SVM classifier given the estimated instance labels. It can be shown that using this approach with the MIMN

[2]In the experiments of this paper, we use cross-validation on the values $K = 3$, $K = 5$, and $K = 10$ to roughly estimate this parameter.

model leads to an algorithm very similar to mi-SVM (Andrews et al., 2002).

However, we use the non-convex cutting plane method (Do and Artières, 2009) to directly solve the optimization problem in (13). This method is proved to converge to a local optimum, unlike the heuristic iterative algorithm of mi-SVM, which has no convergence guarantee. The non-convex cutting plane method iteratively makes an increasingly accurate piecewise quadratic approximation of the objective function. At each iteration, a new linear cutting plane is obtained via the subgradient of the objective function and added to the piecewise quadratic approximation. To use this algorithm, the principal issue is to compute the subgradients $\partial_{\mathbf{w}}\mathcal{L}^n(\mathbf{w})$ and $\partial_{\mathbf{w}}\mathcal{R}^n(\mathbf{w})$. To this end, we need to know the subgradient of the network potential function, i.e., $\partial_{\mathbf{w}} f_{\mathbf{w}}(\mathbf{X}, \mathbf{h}, y)$.

It is simple to show that

$$\partial_{\mathbf{w}} f_{\mathbf{w}}(\mathbf{X}, \mathbf{h}, y) = \Psi(\mathbf{X}, \mathbf{h}, y), \tag{14}$$

where $\Psi(\mathbf{X}, \mathbf{h}, y) = \left[\sum_i \psi_I(\mathbf{x}_i, h_i)^\top, \Psi_C(\mathbf{h}, y)^\top\right]^\top$. Using equations (13) and (14), it can be shown that $\partial_{\mathbf{w}}\mathcal{L}^n(\mathbf{w}) = \Psi(\mathbf{X}^n, \mathbf{h}^\star, y^\star)$, where $(\mathbf{h}^\star, y^\star)$ is the solution to the inference problem:

$$\max_y \max_{\mathbf{h}} \left(\Delta(y, y^n) + f_{\mathbf{w}}(\mathbf{X}^n, \mathbf{h}, y)\right). \tag{15}$$

This inference problem can be solved using the algorithm in 4.1. In summary, for $y = 1$ and $y = -1$ we find $\mathbf{h}$ by doing inference on the resulting graphical model (which has cardinality-based clique potential). Then, the y with the highest value gives the predicted bag label $y^\star$.

In the same way, it can be shown that $\partial_{\mathbf{w}}\mathcal{R}^n(\mathbf{w}) = \Psi(\mathbf{X}^n, \mathbf{h}^\star, y^n)$, where $\mathbf{h}^\star$ is the solution to the inference problem:

$$\max_{\mathbf{h}} f_{\mathbf{w}}(\mathbf{X}^n, \mathbf{h}, y^n). \tag{16}$$

5 EXPERIMENTS

In this section we describe the evaluation of our MIL models. First, the proposed models are evaluated on MIL benchmark data sets to demonstrate they can achieve state of the art performance on standard datasets. Next, we evaluate the models on a challenging cyclist helmet recognition dataset, and show that flexibility in the portion of positives in a bag can lead to improved classification accuracy.

5.1 BENCHMARK DATA SETS

We evaluate the MIL models on five well-known MIL datasets. These benchmark data sets are the *Elephant*, *Fox*, *Tiger* image data sets (Andrews et al., 2002) and *Musk1* and *Musk2* drug activity prediction data sets (Dietterich et al., 1997). In the image data sets, each bag represents an image, and the instances inside the bag represent 230-D feature vectors of different segmented blobs of the image. These data sets contain 100 positive and 100 negative bags. In the MUSK data sets, each bag describes a molecule, and the instances inside the bag represent 166-D feature vectors of the low-energy configurations of the molecule. Musk1 has 47 positive bags and 45 negative bags with about 5 instances per bag. Musk2 has 39 positive bags and 63 negative bags with variable number of instances in a bag, ranging from 1 to 1044 (average 64 instances per bag). Note that in all experiments of this section, we have used normalized data sets, which are obtained by scaling the features of the original data sets[3] to the range $[0, 1]$.

The 10-fold averaged classification accuracies for the MIMN model on different data sets are shown in Table 1. At each trial, we run the non-convex cutting plane algorithm with all the learning weights initialized to 0, roughly optimized λ, and at most 300 iterations. This table also includes the classification results with different kernel feature maps. For these data sets (especially Musk1 and Musk2), non-linear kernels are commonly used for SVM-like algorithms. For example, in (Andrews et al., 2002) mi-SVM and MI-SVM are trained on Musk1 and Musk2 data sets by RBF kernels. Or in (Ray and Craven, 2005) and (Bunescu and Mooney, 2007) quadratic kernels have shown successful classification results. Since our algorithm works with linear kernels, we exploit the idea of kernel feature maps. We investigate the performance of quadratic features in addition to the feature maps proposed in (Vedaldi and Zisserman, 2012) for homogeneous kernels: intersection, χ^2, and Jensen-Shannon.

Table 1: MIMN classification accuracy with different kernel functions. The best results are marked in bold face.

Method	Elephant	Fox	Tiger	Musk1	Musk2
$\text{MIMN}_{\text{Linear}}$	85.5	62.5	**87.0**	78.3	77.6
$\text{MIMN}_{\text{Quadratic}}$	82.50	**64.0**	**87.0**	85.9	81.9
$\text{MIMN}_{\text{Intersection}}$	**89.0**	59.0	85.5	**86.1**	89.5
MIMN_{χ^2}	87.0	60.0	84.0	84.1	**90.3**
$\text{MIMN}_{\text{Jensen-Shannon}}$	86.0	59.0	84.5	83.7	87.4

Now, we compare the best of MIMN with state-of-the-

[3]The original data sets are available online at `http://www.cs.columbia.edu/~andrews/mil/datasets.html`.

art MIL methods in Table 2[4]. The performance of the methods varies depending on the data set. However, MIMN is always among the best methods. More specifically, it achieves the best accuracy in the Elephant, Fox, Tiger and Musk2 data sets.

Note that the competing methods miGraph and MIGraph (Zhou et al., 2009) treat the instances as non-i.i.d samples and model correlations among the bags – this incorporates different information into the model, which is not directly present in our approach.

Next, the results of the experiments with the RMIMN model are presented in Table 3. It can be observed that for the image data sets RMIMN cannot improve MIMN significantly. However, for Musk1 and Musk2 substantial performance gains can be made. The reason might be that in an image usually one of the segments is the true segment (positive instance). So, the prior information, at least one of the instances is positive, is likely sufficient. However, in the Musk data sets, more than one configuration of a molecule might be positive. In fact, it has been previously reported (Gehler and Chapelle, 2007; Hajimirsadeghi and Mori, 2012) that the Musk data sets contain many positive instances in each positive bag. This experiment shows that our graphical approach to MIL allows for exploring different levels of ambiguity in the bags in order to enhance classification accuracy.

Table 3: RMIMN classification accuracy with different ρ values, compared with MIMN. All results are based on linear kernel functions.

Method	ρ	Elephant	Fox	Tiger	Musk1	Musk2
MIMN	-	85.5	62.5	87.0	78.3	77.6
RMIMN	0.1	85.5	62.0	85.5	80.4	79.9
RMIMN	0.2	83.5	61.0	85.0	88.1	82.8
RMIMN	0.3	84.0	56.5	83.5	83.9	88.6
RMIMN	0.4	83.5	60.0	83.0	82.7	84.6
RMIMN	0.5	83.5	59.5	83.5	86.0	86.6
RMIMN	0.6	84.0	59.5	84.0	85.8	86.3
RMIMN	0.7	84.5	58.0	84.0	85.0	84.5
RMIMN	0.8	84.0	57.5	83.5	83.8	82.6
RMIMN	0.9	85.0	61.0	83.5	83.8	82.8
RMIMN	1.0	87.5	62.5	84.5	89.1	84.8

Finally, the results of the experiments with the GMIMN model are provided in Table 4. We evaluate the performance of this model with $K = 10$ weighted segments. It can be observed that although GMIMN gets very weak prior information on the notion of positive bags, by learning the levels of ambiguity in data it outperforms MIMN in most cases.

[4]Note that the reported results for some other methods (e.g. mi-SVM and MI-SVM) on different data sets are also based on the most successful kernels.

Table 4: GMIMN classification accuracy, compared with MIMN. All results are based on linear kernel functions.

Method	Elephant	Fox	Tiger	Musk1	Musk2
MIMN	85.5	62.5	87.0	78.3	77.6
GMIMN($K = 10$)	89.0	61.5	86.5	87.1	81.4

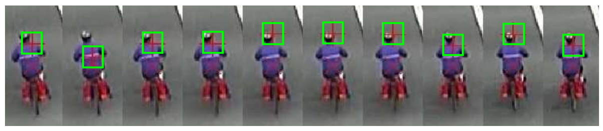


Figure 2: Cyclist helmet classification – is she wearing helmet? how many positives are in this bag? An automatic cyclist detector/tracker is run, with head position estimate in green rectangle. Data instances are features defined on the head position estimates, bags aggregate these over a track.

5.2 CYCLIST HELMET RECOGNITION

The previous experiments show that the proposed method is comparable to the state-of-the-art on standard datasets. However, those datasets exhibit limited ambiguity in positive bags. We now show that for more complex situations, our framework can effectively discover the ambiguity in positive bags. In this section, we use our proposed models to address a video classification task. This problem is illustrated in Figure 2. Given an automatically-obtained cyclist trajectory, we must determine whether the cyclist is wearing a helmet or not. One can treat this as a MIL problem – each frame is an instance, and the trajectory forms a bag. The bag (trajectory) should be classified as containing a helmet-wearing cyclist or not. However, the standard MIL or traditional supervised learning approaches (e.g. classify each instance and majority vote) cannot easily handle this problem. Because of imperfection in tracking, it is unlikely that all the instances in a positive bag are truly positive – some will not be well centered on the cyclist's head due to jitter, regardless of the tracker used. Traditional supervised learning would have many corrupted positive instances of helmet-wearing cyclists. Standard MIL would not make full use of the training data, since each track would very likely have more than one positive instance.

5.2.1 Experimental Setup

We work with cyclist trajectories automatically extracted from video data. The data are collected for a busy 4-legged intersection with vehicles, pedestrians, and cyclists, over a two-day period. Kanade-Lucas-Tomasi feature tracking and trajectory clustering are used to extract moving objects. These clusters are

Table 2: Comparison between state-of-the-art MIL methods. The best and second best results are highlighted in bold and italic face respectively.

Method	Elephant	Fox	Tiger	Musk1	Musk2
MIMN	**89**	**64**	**87**	86	**90**
mi-SVM (Andrews et al., 2002)	82	58	79	87	84
MI-SVM (Andrews et al., 2002)	81	59	84	78	84
MI-Kernel (Gärtner et al., 2002)	84	60	84	88	*89*
MIRealBoost (Hajimirsadeghi and Mori, 2012)	83	*63*	73	**91**	77
MIForest (Leistner et al., 2010)	84	**64**	82	85	82
MILES (Chen et al., 2006)	81	62	80	88	83
AW-SVM (Gehler and Chapelle, 2007)	82	**64**	83	86	84
AL-SVM (Gehler and Chapelle, 2007)	79	*63*	78	86	83
EM-DD (Zhang and Goldman, 2002)	78	56	72	85	85
SVR-SVM (Li and Sminchisescu, 2010)	85	80	*63*	88	85
MIGraph (Zhou et al., 2009)	85	61	82	*90*	**90**
miGraph (Zhou et al., 2009)	*87*	62	*86*	*90*	**90**

then automatically classified (vehicle, pedestrian, cyclist) by analyzing speed profiles (e.g. the pedalling cadence).

We chose a dataset of 24 cyclist tracks for our experiments – 12 wearing helmets and 12 not. The head location is estimated using background subtraction upon the tracks. Samples of tracking the cyclists' heads in the videos are shown in Figure 3. We describe each frame of a track using texton histograms (Malik et al., 2001) in a region of size 20×20 around the head position (chosen after empirically examining other features). We report the results of helmet classification using leave-one-out cross-validation on this dataset.

We introduce a MIL approach to classify sequences. Each video is treated as a bag of frames represented by instances, and we use the proposed models in Section 3 to classify the bags. We also compare this approach with non-MIL methods. In the non-MIL approach, all frames from positive and negative training videos are put together and labelled according to their video labels. Next, a standard SVM classifier (Chang and Lin, 2011) is trained and used to predict each frame label of the test videos. Finally, the bag label is predicted by one of the following criteria:

- SVM-AtLeastOne: The bag label is positive if at least one of the instance labels is positive.
- SVM-Majority: The bag label is specified by the majority voting of the instance labels.

5.2.2 Experimental Results

The average classification accuracy of each method is shown in Table 5. We include mi-SVM as an additional baseline. The results of the RMIMN model have been provided with different ρ values.

Table 5: Results of the experiments on cyclist helmet classification problem.

Method	Accuracy %
SVM-AtLeastOne	58.33
SVM-Majority	79.17
mi-SVM	62.50
MIMN	58.33
GMIMN ($K = 5$)	87.50
RMIMN ($\rho = 0.1$)	79.17
RMIMN ($\rho = 0.2$)	83.33
RMIMN ($\rho = 0.3$)	91.67
RMIMN ($\rho = 0.4$)	87.50
RMIMN ($\rho = 0.5$)	91.67
RMIMN ($\rho = 0.6$)	91.67
RMIMN ($\rho = 0.7$)	87.50
RMIMN ($\rho = 0.8$)	87.50
RMIMN ($\rho = 0.9$)	83.33
RMIMN ($\rho = 1.0$)	66.67

It can be observed that the classification accuracy of SVM-AtLeastOne, MIMN, and mi-SVM are quite low. This shows that the traditional classification approach and MIL definition (used in SVM-AtLeastOne, MIMN, and mi-SVM) are very inefficient in this problem. The traditional MIL definition (i.e., at least one instance of a positive bag is positive) fails because it is very likely that at least one of the instances in a negative bag is classified as positive, and consequently most of the negative bags are assigned positive labels. This problem is due to the imperfection in the classifier and low-quality visual representation of the cyclist's head

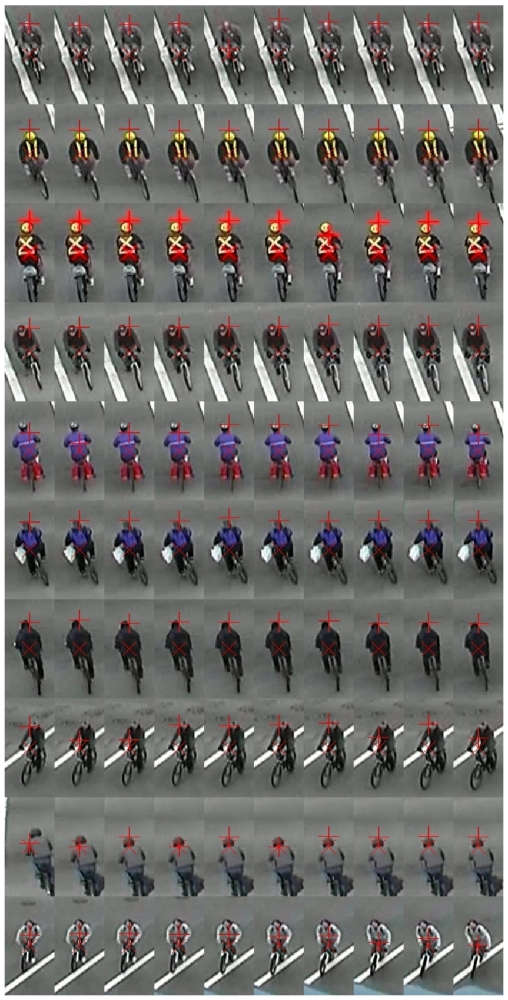


Figure 3: Samples of tracking the cyclists' heads in the videos. Red + shows automatic head position estimate.

in the video. However, it is clearly evident that SVM-Majority, RMIMN (with most ρ values), and GMIMN are more robust to these defects. The results show that RMIMN with $\rho = 0.3$, 0.5, and 0.6 outperform all the other methods. Also, it is shown that GMIMN has good performance, learning the MIL definition properly without any prior knowledge of ambiguity level (e.g., parameter ρ) and classifying the videos successfully.

6 CONCLUSION

We proposed a novel graphical framework for MIL based on Markov networks and max-margin discriminative training. This framework is flexible and can model the traditional MIL definition as well as more general MIL definitions. Thus, it is more robust to the amount of ambiguity (i.e. true positive instances) in the bags. Especially, it can be helpful in vision applications which exhibit imperfect annotation or ambiguous feature representations. For training the proposed models, we formulated the learning process as a max-margin optimization problem.

Experiments on MIL benchmark data sets showed that the proposed algorithm is comparable with state-of-the-art MIL methods. In addition, it was verified that learning and encoding the degree of ambiguity in the classifier can influence the accuracy of classification. We used the proposed framework for classifying cyclist trajectories. This is a challenging problem, where the traditional supervised learning and traditional MIL definitions fail. However, the RMIMN and GMIMN models enhance classification performance by finding more general and robust MIL definitions and mining the degree of ambiguity.

The proposed graphical framework is flexible and can be easily extended or modified. For example, it can be modified to define a bag margin based on the most positive instance of the bag, e.g. MI-SVM (Andrews et al., 2002). It can be also extended for multi-class classification. In addition, more potential functions can be defined between the network nodes. For example, a potential function can be added between a bag-level feature vector and the bag label, or new potential functions can be defined over neighbouring instance labels to treat the instances as non-i.i.d. samples. Finally, this framework could be adapted for individual classification from group statistics, and applied to tasks such as privacy-preserving data mining, election results analysis, spam and fraud detection (Rueping, 2010).

Acknowledgements

This work was supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC) and BCFRST NRAS Research Team Program.

References

S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *NIPS*, 2002.

B. Babenko, M.H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Computer Vision and Pattern Recognition (CVPR)*, 2009.

R.C. Bunescu and R.J. Mooney. Multiple instance learning for sparse positive bags. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 105–112. ACM, 2007.

C.C. Chang and C.J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

Y. Chen, J. Bi, and J.Z. Wang. Miles: Multiple-instance learning via embedded instance selection. *T-PAMI*, 28(12):1931–1947, 2006.

Thomas Deselaers and Vittorio Ferrari. A conditional random field for multiple-instance learning. 2010.

T.G. Dietterich, R.H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2): 31–71, 1997.

T.M.T. Do and T. Artières. Large margin training for hidden markov models with partially observed states. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 265–272. ACM, 2009.

L. Duan, W. Li, I. Tsang, and D. Xu. Improving web image search by bag-based re-ranking. *IEEE Transactions on Image Processing*, 20(11): 3280–3290, 2011.

P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9):1627–1645, 2010.

T. Gärtner, P.A. Flach, A. Kowalczyk, and A.J. Smola. Multi-instance kernels. In *Proceedings of the 19th International Conference on Machine Learning (ICML)*, pages 179–186, 2002.

P. Gehler and O. Chapelle. Deterministic annealing for multiple-instance learning. In *AISTATS*, 2007.

R. Gupta, A.A. Diwan, and S. Sarawagi. Efficient inference with cardinality-based clique potentials. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 329–336. ACM, 2007.

H. Hajimirsadeghi and G. Mori. Multiple instance real boosting with aggregation functions. In *Proceedings of the 21st International Conference on Pattern Recognition*, 2012.

H. Kueck and N. de Freitas. Learning about individuals from group statistics. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2005.

C. Leistner, A. Saffari, and H. Bischof. Miforests: Multiple-instance learning with randomized trees. In *European Conference on Computer Vision (ECCV)*, 2010.

F. Li and C. Sminchisescu. Convex multiple-instance learning by estimating likelihood ratio. *Advances in Neural Information Processing Systems (NIPS)*, pages 1360–1368, 2010.

W. Li, L. Duan, D. Xu, and I.W.H. Tsang. Text-based image retrieval using progressive multi-instance learning. In *International Conference on Computer Vision (ICCV)*, pages 2049–2055. IEEE, 2011.

J. Malik, S. Belongie, T. K. Leung, and J. Shi. Contour and texture analysis for image segmentation. *International Journal of Computer Vision*, 43(1):7–27, 2001.

O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. *Advances in Neural Information Processing Systems (NIPS)*, pages 570–576, 1998.

N. Quadrianto, A. Smola, T. Caetano, and Q. Le. Estimating labels from label proportions. *The Journal of Machine Learning Research*, 10:2349–2374, 2009.

S. Ray and M. Craven. Supervised versus multiple instance learning: An empirical comparison. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 697–704. ACM, 2005.

S. Rueping. Svm classifier estimation from group probabilities. In *Proc. of the 27th Int. Conf. on Machine Learning (ICML)*, 2010.

Daniel Tarlow, Kevin Swersky, Richard S Zemel, Ryan Prescott Adams, and Brendan J Frey. Fast exact inference for recursive cardinality models. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2012.

A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3): 480–492, 2012.

P. Viola, J. Platt, and C. Zhang. Multiple instance boosting for object detection. In *NIPS*, 2006.

Jonathan Warrell and Philip HS Torr. Multiple-instance learning with structured bag models. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 369–384. Springer, 2011.

Q. Zhang and S.A. Goldman. Em-dd: An improved multiple-instance learning technique. *Advances in Neural Information Processing Systems (NIPS)*, 14: 1073–1080, 2002.

Z.H. Zhou, Y.Y. Sun, and Y.F. Li. Multi-instance learning by treating instances as non-iid samples. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 1249–1256. ACM, 2009.

Unsupervised Learning of Noisy-Or Bayesian Networks

Yoni Halpern, David Sontag
Department of Computer Science
Courant Institute of Mathematical Sciences
New York University

Abstract

This paper considers the problem of learning the parameters in Bayesian networks of discrete variables with known structure and hidden variables. Previous approaches in these settings typically use expectation maximization; when the network has high treewidth, the required expectations might be approximated using Monte Carlo or variational methods. We show how to avoid inference altogether during learning by giving a polynomial-time algorithm based on the method-of-moments, building upon recent work on learning discrete-valued mixture models. In particular, we show how to learn the parameters for a family of bipartite noisy-or Bayesian networks. In our experimental results, we demonstrate an application of our algorithm to learning QMR-DT, a large Bayesian network used for medical diagnosis. We show that it is possible to fully learn the parameters of QMR-DT even when only the findings are observed in the training data (ground truth diseases unknown).

1 Introduction

We address the problem of unsupervised learning of the parameters of bipartite noisy-or Bayesian networks. Networks of this form are frequently used models for expert systems and include the well-known Quick Medical Reference (QMR-DT) model for medical diagnosis (Miller *et al.* , 1982; Shwe *et al.* , 1991).

Given that QMR-DT is one of the most well-studied noisy-or Bayesian networks, we use it as a running example for the type of network that we would like to provably learn. It is a large bipartite network, describing the relationships between 570 binary disease variables and 4,075 binary symptom variables using 45,470 directed edges. It was laboriously assembled based on information elicited from experts and represents an example of a network that captures (at least some of) the complexities of real-world medical diagnosis tasks.

Learning these parameters is important. Both the structure and the parameters of the QMR-DT model were manually specified, taking over 15 person-years of work (Miller *et al.* , 1982). Each disease took one to two weeks of full-time effort, involving in-depth review of the medical literature, to incorporate into the model. Despite this effort, the original INTERNIST-1/QMR model still lacked an estimated 180 diseases relevant to general internists (Miller *et al.* , 1986). Furthermore, model parameters such as the priors over the diseases can vary over time and location.

Although it is often possible to extract symptoms or findings from unstructured clinical data, obtaining reliable ground truth for a patient's underlying disease state is much more difficult. Often all we have available are noisy and biased estimates of the patient's disease state in the form of billing or diagnosis codes and free text. We can, however, treat these noisy labels as additional findings (for training) and perform unsupervised learning. The ability to learn parameters from unlabeled data could make models like QMR-DT much more widely applicable.

Exact inference in the QMR-DT network is known to be intractable (Cooper, 1987), so it would be expected to resort to expectation-maximization techniques using approximate inference in order to learn the parameters of the model (Jaakkola & Jordan, 1999; Šingliar & Hauskrecht, 2006). However, these methods can be computationally costly and are not guaranteed to recover the true parameters of the network even when presented with infinite data drawn from the model.

We give a polynomial-time algorithm for provably learning a large family of bipartite noisy-or Bayesian networks. It is important to note that this method does not extend to all bipartite networks. It does not work on certain densely connected structures. We provide a criterion based on the network structure to determine whether or not the network is learnable by our algorithm. Though the algorithm is limited, the family of networks for which we can learn parameters is certainly non-trivial.

Our approach is based on the method-of-moments, and builds upon recent work on learning discrete-valued mixture models (Anandkumar *et al.* , 2012c; Chang, 1996; Mossel & Roch, 2005). We assume that the observed data is drawn independently and identically distributed from a model of known structure and unknown parameters, and show that we can accurately and efficiently recover those parameters with high probability using a reasonable number of samples. Making these additional assumptions allows us to circumvent the hardness of maximum likelihood learning.

Our parameter learning algorithm begins by finding triplets of observed variables that are *singly-coupled*, meaning that they are marginally mixture models. After learning the parameters involving these, we show how one can *subtract* their influence from the empirical distribution, which then allows for more parameters to be learned. This process continues until no new parameters can be learned. Surprisingly, we show that this simple algorithm is able to learn almost all of the parameters of the QMR-DT structure. Finally, we study the identifiability of the learning problem with hidden variables and show that even in dense networks, the true model is often identifiable from third-order moments. Our identifiability results suggest that the final parameters of QMR-DT can be learned with a grid search over a single parameter.

We see the significance of our work as presenting one of the first polynomial-time algorithms for learning a family of discrete-valued Bayesian networks with hidden variables where exact inference on the hidden variables is intractable. We believe that our algorithm will be of practical interest in applications (such as medical diagnosis) where prior knowledge can be used to specify the Bayesian network structure involving the hidden variables and the observed variables.

2 Background

We consider bipartite noisy-or Bayesian networks with n binary latent variables, $D = \{D_1, D_2, ..., D_n\}, D_i \in \{0,1\}$, and m observed binary variables, $S = \{S_1, S_2, ..., S_m\}, S_i \in \{0,1\}$. Continuing with the medical diagnosis example, we refer to the latent variables as *diseases* and the observed variables as *symptoms*. The edges in the model are directed from the latent diseases to the observed symptoms. We assume that the diseases are never observed, neither at training nor test time, and show how to recover the parameters of the model in an unsupervised manner.

By using a noisy-or conditional distribution to model the interactions from the latent variables to the observed variables, the entire Bayesian network can be parametrized by $n \times m + n + m$ parameters. These parameters consist of *prior* probabilities on the diseases $\Pi = \{p_1, p_2, ..., p_n\}$, *failure* probabilities between diseases and symptoms, $F = \{\vec{f_1}, \vec{f_2}, ...\vec{f_n}\}$, where each $\vec{f_i}$ is a vector of size m, and noise (or leak) probabilities $\vec{\nu} = \{\nu_1, ...\nu_m\}$. An equivalent formulation includes the noise in the model by introducing a single 'noise' disease, d_0, which is present with probability $p_0 = 1$ and has failure probabilities $\vec{f_0} = 1 - \vec{\nu}$.

Observations are sampled from the noisy-or network by the following generative process:
— The set of present diseases is drawn according to Bernoulli(Π).
— For each present disease D_i, the set of active edges $\vec{a}_i$, is drawn according to Bernoulli($1 - \vec{f_i}$).
— The observed value of the j^{th} symptom is then given by $s_j = \bigcup_i a_{i,j}$ (this part is deterministic).

While the network can be described generally as being fully connected, in practice many of the diseases have zero probability of generating many of the symptoms (ie. fail with probability 1). The Bayesian network only has an edge between disease D_i and symptom S_j if $f_{i,j} < 1$. As we explain in Section 3, our ability to learn parameters will depend on the particular sparsity pattern of these edges.

The marginal distribution over a set of symptoms, $\mathcal{S}$, in the noisy-or network has the following form:

$$p(\mathcal{S}) = \sum_{\{D\}} \prod_{i=1}^{n} p(d_i) \prod_{j \in \mathcal{S}} p(s_j|D), \tag{1}$$

where $\{D\}$ is the set of 2^n configurations of the disease variables $\{d_1, ..., d_n\}$. The disease priors are given by $p(d_i) = p_i^{d_i}(1-p_i)^{1-d_i}$, and the conditional distribution of the symptoms by a noisy-or distribution:

$$p(s_j|D) = \Big(1 - f_{0,j} \prod_{i=1}^{n} f_{i,j}^{d_i}\Big)^{s_j} \Big(f_{0,j} \prod_{i=1}^{n} f_{i,j}^{d_i}\Big)^{1-s_j} \tag{2}$$

The algorithms described in this paper make substantial use of sets of moments of the observed variables. The first moment that will be important is the joint distribution over a set of symptoms, $\mathcal{S}$, which we call $T_{\mathcal{S}}$. $T_{\mathcal{S}}$ is a $|\mathcal{S}|^{th}$ order tensor where each dimension is of size 2. For a set of symptoms $\mathcal{S} = (S_a, S_b, S_c)$ the elements of $T_{\mathcal{S}}$ are defined as: $T_{\mathcal{S}(s_a,s_b,s_c)} = p(S_a = s_a, S_b = s_b, S_c = s_c)$. Throughout the paper we will make use of sets of at most three variables, so the joint distributions are of maximal size $2 \times 2 \times 2$.

We also make use of the negative moment of a set of symptoms $\mathcal{S}$, which we denote as $\bar{M}_{\mathcal{S}}$, defined as the marginal probability of observing *all* of the symptoms in $\mathcal{S}$ to be absent. The negative moments of $\mathcal{S}$ have the following compact form:

$$\bar{M}_{\mathcal{S}} \equiv p(\bigcap_{S_j \in \mathcal{S}} S_j = 0) = \prod_{i=0}^{n} \Big(1 - p_i + p_i \prod_{S_j \in \mathcal{S}} f_{i,j}\Big) \quad (3)$$

The form of Eq. 3 makes it clear that the parameters associated with each parent are all grouped together in a single term, which we call the *influence* of disease D_i on symptoms $\mathcal{S}$. Define this influence term to be $I_{i,\mathcal{S}} \equiv 1 - p_i + p_i \prod_{S_j \in \mathcal{S}} f_{i,j}$. Using this, we rewrite Eq. 3 using influences as $\bar{M}_{\mathcal{S}} = \prod_{i=0}^{n} I_{i,\mathcal{S}}$. This formulation is found in Heckerman (1990) and provides a compact form that makes it easy to take advantage of the noisy-or properties of the network.

2.1 Related Work

The problem of inference in bipartite noisy-or networks with fixed parameters has been studied and exact inference in large models like the QMR-DT model is known to be intractable (Cooper, 1987). The Quickscore formulation by Heckerman (1990) takes advantage of the noisy-or parameterization to give an exact inference algorithm that is polynomial in the number of negative findings but still exponential in the number of positive findings.

Any expectation maximization (EM) approach to learning the network parameters must contend with the computational complexity of inference in these models. Many approximate inference strategies have been developed, notably Jaakkola & Jordan (1999) and Ng & Jordan (2000). The closest related work to our paper is by Šingliar & Hauskrecht (2006), who give a variational EM algorithm for unsupervised learning of the parameters of a noisy-or network. We will use their algorithm as a baseline in our experimental results. Importantly, variational EM algorithms do not have consistency guarantees.

Kearns & Mansour (1998) develop an inference-free approach which is guaranteed to learn the exact structure and parameters of a noisy-or network under specific identifiability assumptions by performing a search over network structures. In order to achieve their results, they impose strong constraints such as identical priors on all of the parents. Their structure learning algorithm is exponential in the maximal in-degree of the symptom nodes, which for QMR-DT is 570. More importantly, the overall approach relies on the model family having a property called "unique polynomials", closely related to the question of identifiability, but which is left mostly uncharacterized in their paper. It is not clear whether their algorithm can be modified to take advantage of a known structure. As such, no existing method is sufficient for learning the parameters of a large network like the QMR-DT network.

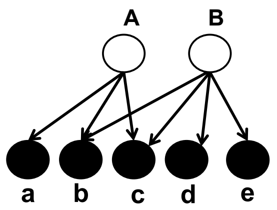


Figure 1: A small noisy-or network. The triplets (b,d,e) and (c,d,e) are both singly-coupled by B. The presence of disease B prevents (a,b,c) from being singly-coupled. However, after learning the parameters of disease B, we can subtract off its influence, leaving (a,b,c) singly-coupled.

Spectral approaches to learning mixture models originated with Chang's spectral method (Chang 1996; analyzed in Mossel & Roch 2005). These methods have been successfully applied to learning discrete mixture models and hidden Markov models (Anandkumar *et al.* , 2012c), as well as continuous admixture models such as latent Dirichlet allocation (Anandkumar *et al.* , 2012b). In recent work, these have been generalized to a large class of linear latent variable models (Anandkumar *et al.* , 2012d). However, the noisy-or model is not linear, making it non-trivial to apply these methods that rely on linearity of expectation to relate the general formula for observed moments to a low rank matrix or tensor decomposition.

3 Parameter Learning with Known Structure

In this section we present a learning algorithm that takes advantage of the known structure of a noisy-or network in order to learn parameters using only low-order moments. We first identify singly-coupled triplets, which are marginally mixture models and therefore we can learn their parameters. Once some parameters of the network are learned, we make adjustments to the observed moments, subtracting off the influence of some parents, essentially removing them from the network, making more triplets singly-coupled (illustrated in Figure 1). Algorithm 1 outlines the parameter learning procedure.

We discuss the running time in Section 3.4. The clean up procedure is not part of the main algorithm and may increase the runtime to exponential, depending on the configuration of the network of remaining parameters at the end of the main algorithm. We present it because it allows us to extend the algorithm to learn

Algorithm 1 Learn Parameters

```
Inputs: A bipartite noisy-or network structure with unknown parameters $F, \Pi$. $N$ samples from the network.
Outputs: Estimates of $F$ and $\Pi$.
– Main Routine
 1: unknown = $\{f_{i,j} \in F\} \cup \{p_i \in \Pi\}$
 2: knowns = $\{\}$
 3: while not converged do
 4:     learned = $\{\}$
 5:     for all $f_{i,a}$ in unknown_parameters do
 6:         for all $(S_b, S_c)$, siblings of $S_a$ do
 7:             Parents = parents of $(S_a, S_b, S_c)$
 8:             knownParents = All $D_k$ in Parents for which $f_{k,a}, f_{k,b}$ and $f_{k,c}$ are known.
 9:             Remove knownParents from the graph.
10:             if $(S_a, S_b, S_c)$ are singly-coupled (Def. 1) then
11:                 Form joint distribution $T_{a,b,c}$
12:                 for all $D_k$ in knownParents do
13:                     $T_{a,b,c}$ = RemoveInfluence($T_{a,b,c}$, $D_k$) (Section 3.2)
14:                 end for
15:                 Learn $p_i, f_{i,a}, f_{i,b}, f_{i,c}$. (Eq. 4)
16:                 unknown = unknown - $(p_i, f_{i,a}, f_{i,b}, f_{i,c})$.
17:                 learned = learned $\cup (p_i, f_{i,a}, f_{i,b}, f_{i,c})$
18:             end if
19:             Add back knownParents to the graph.
20:         end for
21:     end for
22:     known = known $\cup$ learned
23:     Converge if no new parameters are learned.
24: end while
25: Learn noise parameters (Eq. 5).
– Clean up
 1: Check identifiability of remaining parameters with third-order moments and use clean up procedure to learn remaining parameters. (Section 3.3)
```

the QMR-DT network which has a very simple network of remaining parameters after running the main algorithm to completion.

The algorithm can be further optimized by precomputing and storing dependencies between triplets (i.e., triplet A can be learned after triplet B is learned) to avoid repeated searches for singly-coupled triplets. The algorithm is also greedy in that it learns each failure parameter $f_{i,j}$ with the first suitable triplet it encounters. A more sophisticated version would attempt to determine the best triplet to learn $f_{i,j}$ with high confidence, which we do not explore in this paper.

The following sections go into more detail on the various steps of the algorithm, and assume that we have access to the exact moments (i.e., infinite data). In Section 3.4 we show that the error incurred by using sample estimates of the expectations is bounded.

3.1 Learning Singly-coupled Symptoms

The condition that we require to learn the parameters is that the observed variables be *singly-coupled*:

Definition 1. *A set of symptoms, $\mathcal{S}$ is* singly-coupled *by parent D_i if D_i is a parent of S_j for all $S_j \in \mathcal{S}$ and there is no other parent, $D_k \in \{D_1, ..., D_n\}$, such that D_k is a parent of at least two symptoms in $\mathcal{S}$.*

The intuition behind using singly coupled symptoms is they can be viewed locally as mixture models with two mixture components corresponding to the states of the coupling parent. For example, in Figure 1, (b, d, e) and (c, d, e) form singly-coupled triplets coupled by disease B. Their observations are independent conditioned on the state of B. The noise disease, D_0, does not have to be considered here since it is present with probability 1, and so its state is always observed. Thus, the noise parent can never act as a coupling parent.

Observing that the singly-coupled condition locally creates a binary mixture model, we conclude that we can learn the noisy-or parameters associated with a singly-coupled triplet by using already existing methods for learning 3-view mixture models from the third-order moment $T_{a,b,c}$. While the general method of learning multi-view mixture models described in Anandkumar *et al.* (2012a) would suffice, we employ a simpler method (given in Algorithm 2) applicable to mixture models of binary variables based on a tensor decomposition described in Berge (1991). This procedure uniquely decomposes $T_{a,b,c}$ into two rank-1 tensors which describe the conditional distributions of the symptoms conditioned on the state of the parent.

The tensor decomposition returns the prior probabilities of the parent states and the probabilities of the children conditioned on the state of the parent. Ambiguity in the labeling of the parent states is avoided since for noisy-or networks $p(S_j = 0|D_i = 0) > p(S_j = 0|D_i = 1)$. To obtain the noisy-or parameters, we observe that the prior for the disease is simply given by the mixture prior, and the failure probability $f_{i,j}$ between the coupling disease D_i and symptom S_j is the ratio of two conditional probabilities:

$$p_i = p(D_i = 1), \qquad f_{i,j} = \frac{p(S_j = 0|D_i = 1)}{p(S_j = 0|D_i = 0)}. \qquad (4)$$

The noise parameter $f_{0,j}$ is not learned using the above equations since D_0 never acts as a coupling parent. However, once all of the other parameters are learned, the noise parameter simply provides for any otherwise

Algorithm 2 Binary Tensor Decomposition

Input: Tensor T of size $2 \times 2 \times 2$ which is a joint probability distribution over three variables (S_a, S_b, S_c) which are singly-coupled by disease Z.
Output: Prior probability $p(Z = 1)$, and conditional distributions $p(s_a, s_b, s_c|Z = 0)$, $p(s_a, s_b, s_c|Z = 1)$.

```
1: Matrix $X_1 = T_{(0,\cdot,\cdot)}$
2: Matrix $X_2 = T_{(1,\cdot,\cdot)}$
3: $Y_2 = X_2 X_1^{-1}$
4: Find eigenvalues of $Y_2$ using quadratic equation:
5: $\lambda_1, \lambda_2 = \text{roots}(\lambda^2 - \text{Tr}(Y_2)\lambda + \text{Det}(Y_2))$
6: $\vec{u}_1\vec{v}_1^T = (\lambda_1 - \lambda_2)^{-1}(X_2 - \lambda_2 X_1)$
7: $\vec{u}_2\vec{v}_2^T = -(\lambda_1 - \lambda_2)^{-1}(X_2 - \lambda_1 X_1)$
8: Decompose* $\vec{u}_1\vec{v}_1^T$, $\vec{u}_2\vec{v}_2^T$ into $\vec{u}_1$, $\vec{u}_2$, $\vec{v}_1$, $\vec{v}_2$.
9: $\vec{l}_1 = \begin{pmatrix}1 & \lambda_1\end{pmatrix}^T$, $\vec{l}_2 = \begin{pmatrix}1 & \lambda_2\end{pmatrix}^T$
10: $T_1 = \vec{u}_1 \otimes \vec{v}_1 \otimes \vec{l}_1$, $T_2 = \vec{u}_2 \otimes \vec{v}_2 \otimes \vec{l}_2$
11: if $T_{1(0,0,0)} > T_{2(0,0,0)}$ then
12:     swap $T_1$, $T_2$
13: end if
14: $p(Z = 1) = \sum_{i,j,k} T_{2(i,j,k)}$
15: normalize $p(s_a, s_b, s_c|Z = 0) = T_1 / \sum_{i,j,k} T_{1(i,j,k)}$
16: normalize $p(s_a, s_b, s_c|Z = 1) = T_2 / \sum_{i,j,k} T_{2(i,j,k)}$
```

* *To decompose the* 2×2 *matrix* $\vec{u}\vec{v}^T$ *into vectors* $\vec{u}$ *and* $\vec{v}$, *set* $\vec{v}^T$ *to the top row and* $\vec{u}^T = \begin{pmatrix}1 & \frac{(\vec{u}\vec{v}^T)_{(2,2)}}{(\vec{u}\vec{v}^T)_{(1,2)}}\end{pmatrix}$.
–Notation $T = \vec{u} \otimes \vec{v} \otimes \vec{l}$ means that $T_{(i,j,k)} = u_i v_j l_k$.

unaccounted observations, i.e.

$$f_{0,j} = \frac{\bar{M}_j}{\prod_{D_i \in Parents(S_j)} I_{i,j}}. \tag{5}$$

3.2 Adjusting Moments

Consider a triplet (a, b, c) which has a common parent A, but is not singly coupled due to the presence of a parent B shared by b and c (Figure 1). If we wish to learn the parameters involving this triplet and A using the methods described above, we would need to form an adjusted moment, $\tilde{T}_{a,b,c}$ which would describe the joint distribution of (a, b, c) if B did not exist.

The influence of B on variables (b, c) is fully described by the parameters $p_B, f_{B,b}, f_{B,c}$. Thus, if we have estimates for these parameters, we can remove the influence of B to form the joint distribution over (a, b, c) as though B did not exist. This can be seen explicitly in Equation 3. In this form, the influence of each parent, if known, can be isolated and removed from the negative moments with a division operation. Since all the variables are binary, the mapping between the negative moments and the joint distribution is simple and the adjusted joint distribution can be formed from the power set of adjusted negative moments.

This procedure of adjusting moments by removing the influence of parents vastly expands the class of networks whose parameters are fully learnable using the singly-coupled triplet method from Section 3.1. Using these methods, complicated real-world networks such as the QMR-DT network can be learned almost fully. The clean up procedure described in the next section will make it possible to learn the remaining parameters of the QMR-DT network.

3.3 Extensions of the Main Algorithm

Learning with singly-coupled pairs. It is not possible to identify the parameters of a noisy-or model by only looking at singly-coupled pairs. However, once we have information about some of the parameters from looking at triplets, we can use it to find more parameter values by examining pairs. For example, in Figure 1, if p_B and $f_{B,d}$ were learned using the triplet (b, d, e), it would be possible to find $f_{B,c}$ using only the pairwise moment between (c, d). More generally, for a singly-coupled pair of observables (S_i, S_j) coupled by parent D_i, the following linear equation holds and can be used to solve for the unknown $f_{i,k}$ assuming $f_{i,j}$ and p_i are already estimated:

$$\frac{\bar{M}_{\{j,k\}}}{\bar{M}_j \bar{M}_k} = \frac{1 - p_i + p_i f_{i,j} f_{i,k}}{(1 - p_i + p_i f_{i,j})(1 - p_i + p_i f_{i,k})}. \tag{6}$$

Thus, once some parameters have been estimated, singly-coupled pairs provide an alternative to singly-coupled triplets. Extending Algorithm 1 to search for singly-coupled pairs as well as triplets is trivial. For complex networks, using pairs allows us to learn more parameters with fewer adjustment steps.

Clean up procedure. For some Bayesian network structures, after running the main algorithm to completion, we may be left with some unlearned parameters. This occurs because it may be impossible to find enough singly-coupled triplets and pairs.

In these settings, it is natural to ask whether it is possible to uniquely identify the remaining parameters. We use a technique developed by Hsu *et al.* (2012) to show that most fully connected bipartite networks are *locally identifiable*, meaning that they are identifiable on all but a measure zero set of parameter settings. In particular, we use their CheckIdentifiability routine, which computes the Jacobian matrix of the system of moment constraint equations and evaluates its rank at a random setting of the parameters. We start with first-order moments and increase the order until the Jacobian is full rank, which implies that the model is locally identifiable with these moments. When the test succeeds it gives hope that, for all but a very small number of pathological cases, the networks can still be identifiable (up to a trivial relabeling of parents).

		Number of Symptoms						
		1	2	3	4	5	6	7
Number of Hidden Variables	1	-1	-1	3	3	3	3	3
	2	-1	-1	-1	3	3	3	3
	3	-1	-1	-1	-1	3	3	3
	4	-1	-1	-1	-1	4	3	3
	5	-1	-1	-1	-1	-1	3	3
	6	-1	-1	-1	-1	-1	4	3
	7	-1	-1	-1	-1	-1	4	3

Table 1: Identifiability of parameters in fully-connected bipartite networks. Each row represents a number of hidden variables and each column is the number of observed variables. The value at location (i, j) is the number of moments required to make the model identifiable according to the local identifiability criteria of the Jacobian method. E.g., 3rd order moments are needed to learn with a single hidden variable. The value -1 means the model is not identifiable even with the highest possible order moments.

Table 1 summarizes the results on networks with varying number of children. Even for fully connected networks, third-order moments are sufficient to satisfy the local identifiability criteria provided that there are a sufficient number of children.[1]

At this point, we can make progress by relying on the identifiability of the network from third-order moments and doing a grid search over parameter values to find the values that best match the observed third-order moments. For example, consider the network in Figure 2. This could be a sub-network that is left to learn after a number of other parameters have been learned and possibly removed. If we knew the values for the prior p_A and failure probability $f_{A,a}$, then we would learn all of the edges from A and subtract them off using the pairs learning procedure. When we do not know p_A and $f_{A,a}$, we can search over the range of values and choose the values that yield the closest third-order moments to the observed moments.

Significantly, this method of doing a grid search over two parameters can be used no matter how many children are shared by A and B. It only depends on the number of parents whose parameters are not learned. Thus, even if there are a large number of parameters left at the end of the main algorithm, we can proceed efficiently if they belong to a small number of parents. In Section 4.2 we note that in the QMR-DT network, all of the parameters that are left at the end of the main algorithm belong to only two parents and thus can be learned efficiently using the clean up phase.

[1]Third-order moments are also necessary for identifiability. Appendix G of Anandkumar *et al.* (2012a) gives an example of two networks, each with a single latent variable and three observations, that are indistinguishable using only second-order moments.

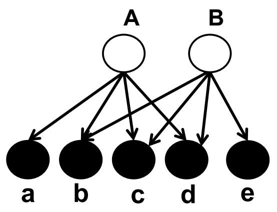


Figure 2: Similar to Figure 1, with the addition of a single edge from A to d. There are now no singly-coupled triplets and learning cannot proceed. In the clean up procedure, we perform a grid search over values for p_A and $f_{A,a}$, use them to learn all of the edges leading to A and then proceed to subtract off the influence of A and learn the edges of B.

3.4 Theoretical Properties

Valid schedule. We call a schedule, describing an order of adjustment and learning steps, *valid* if every learning step operates on a singly-coupled triplet (possibly after adjustment) and every parameter used in an adjustment is learned in a preceding step.

Note that a schedule is completely data independent, and depends only on the structure of the network. Algorithm 1 can be used to find a valid schedule if one exists. A valid schedule can also be used as a certificate of parameter identifiability for noisy-or networks with known structure:

Theorem 1. *If there exists a valid schedule for a noisy-or network, then all parameters are uniquely identifiable using only third-order moments.*

The proof follows from the uniqueness of the tensor decomposition described in Berge (1991).

Computational complexity. We run Algorithm 1 in two passes. In the first pass, we take as input the structure and find a valid schedule. The schedule will use one triplet per edge $f_{ij} \in F$, resulting in at most $|F|$ triplets for which to estimate the moments. Next, we iterate through the data, computing the required statistics. Finally, we do a second pass with the schedule to learn the parameters. The running time without the clean up procedure is $O(nm^2|F|^2 + |F|N)$, where N is the number of samples.

Sample complexity. The parameter learning and adjustments presented above recover the parameters of the network exactly under the assumption that perfect estimates of the moments are available. With finite data sampled i.i.d. from a noisy-or network, the estimates of the moments are subject to sampling noise. In what follows, we bound the error accumulation due to using imperfect estimates of the moments.

Since error accumulates with each learning and adjustment step, we define the *depth* of a parameter θ to be the number of extraction and adjustment steps required to reach the state in which θ can be learned. This depth is defined recursively:

Definition 2. *Denote the parameters used in the adjustment step before learning θ as Θ_{adj}. Depth(θ) $= \max_{\theta_i \in \Theta_{adj}}$ Depth(θ_i) $+ 1$. If no adjustment is needed to learn θ then we say its depth is 0.*

To ensure that parameters are learned with the minimum depth, we construct the schedule in rounds. In round k we learn all parameters that can be learned using parameters learned in previous rounds. We only update the set of known parameters at the end of the round. In this manner we are ensured that at each round, the algorithm learns all of the parameters that can be learned at a given depth.

The sample complexity result will depend on how close the parameters of the model are to 0 or 1. In particular, we define $p_{\min}$, $p_{\max}$ as the minimum and maximum disease priors, and $f_{\max}$ as the maximum failure probability. Additionally, we define $\bar{M}_{\min} = \min_{S_j \in S} \Pr(S_j = 0)$ to be the minimum marginal probability of any symptom being absent.

Our algorithm makes black-box use of an algorithm for learning mixture models of binary variables. In giving our sample complexity result, we abstract the dependence on the particular mixture model learning algorithm as follows:

Definition 3. *Let $f(\bar{M}_{\min}, f_{max}, p_{\max}, p_{\min}, \hat{\delta})$ be a function that represents the multiplicative increase in error incurred by learning the parameters of a mixture model from an estimate $\hat{T}_{a,b,c}$ of the third-order moment $T_{a,b,c}$, such that for all mixture parameters θ,*

$$||\hat{T}_{a,b,c} - T_{a,b,c}||_1 < \hat{\epsilon} \implies |\hat{\theta} - \theta| < f(\bar{M}_{\min}, f_{max}, p_{\max}, p_{\min}, \hat{\delta})\hat{\epsilon}$$

with probability at least $1 - \hat{\delta}$.

Using this, we obtain the sample complexity result (K refers to the maximal in-degree of any symptom):

Theorem 2. *Let Θ be the set of parameters to be learned. Given a noisy-or network with known structure and a valid schedule with some constant maximal depth d, after a number of samples equal to*

$$N = \tilde{O}\Big(\Big(f\Big(\bar{M}_{\min}, f_{max}, p_{\max}, p_{\min}, \frac{\delta}{|\Theta|K^d}\Big)\Big)^{2d+2} \cdot K^{2d}\bar{M}_{\min}^{-6d} \cdot \epsilon^{-2} \cdot \ln(|\Theta|/\delta)\Big)$$

and with probability $1-\delta$, for all $\theta \in \Theta$ Algorithm 1 returns an estimate $\hat{\theta}$ such that $|\hat{\theta} - \theta| < \epsilon$. This holds for $\epsilon < \frac{1}{2} f\Big(\bar{M}_{\min}, f_{max}, p_{\max}, p_{\min}, \frac{\delta}{|\Theta|K^d}\Big)^{-1}\Big(\frac{\bar{M}_{\min}^3}{15K}\Big)$.

The proof consists of bounding the error incurred at each successive operation of learning parameters, using them to adjust the joint distributions, and applying standard sampling error bounds. The multiplicative increase in error with every adjustment and learning step leads to an exponential increase in error when these steps are applied repeatedly in series. The dependence on the maximal in-degree, K, comes from the possibility that in any adjustment step it may be necessary to subtract off the influence of all but one parent of the symptoms in the triplet. The maximum value for ϵ comes from division operations in both the learning and adjustment steps. If ϵ is not sufficiently small then the error can blow up in these steps.

Using the bounds presented for the mixture model learning approach in Anandkumar *et al.* (2012a) gives

$$f(\bar{M}_{\min}, f_{max}, p_{\max}, p_{\min}, \hat{\delta}) \propto \bar{M}_{min}^{-11}(1 - f_{max})^{-10} \cdot (\min\{1 - p_{max}, p_{min}\})^{-2} \cdot \frac{\ln(1/\hat{\delta})}{\hat{\delta}},$$

though these bounds may not be tight. In particularly, the $\frac{1}{\delta}$ dependency in f comes from a randomized step of the learning procedure. For binary variables this step may not be necessary and the $\frac{1}{\delta}$ dependency may be avoidable.

We emphasize that although the sample complexity is exponential in the depth, even complex networks like the QMR-DT network can be shown to have very small maximal depths. In fact, the vast majority of the parameters of the QMR-DT network can be learned with no adjustment at all (i.e., at a depth of 0).

4 Experiments

Our first set of experiments look at parameter recovery in samples drawn from a simple synthetic network with the structure of Figure 1, and compare against the variational EM algorithm of Šingliar & Hauskrecht (2006). This network was chosen because it is the simplest network that requires our method to perform the adjustment procedure to learn some of the parameters.

The comparison is done on a small model to show that that even in this simple case, the variational EM baseline performs poorly. Any larger network could have a subnetwork that looks like the network in Figure 1. In our second set of experiments, we apply our algorithm to the large QMR-DT network and show that our algorithm's performance scales to large models.

4.1 Comparison with (Variational) EM

Our method-of-moments algorithm is compared to variational EM on 64 networks with the structure of

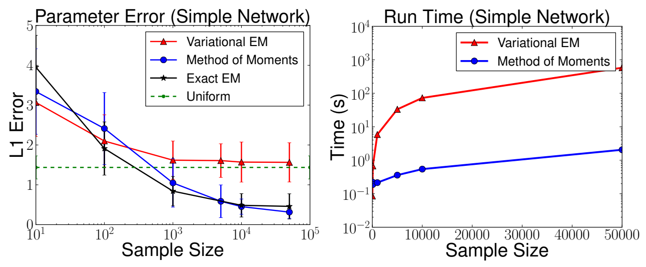


Figure 3: (left) Sum of L1 errors from the true parameters. Error bars show standard deviation from the mean. The dotted line for Uniform denotes the average error from estimating the failures of the noise parent as 1 and failures and priors of all other parents uniformly as 0.5. (right) Run time in seconds of a single run using the network structure from Figure 1 (shown in log scale).

Figure 1 and random parameters. The failure and prior parameters of each network were generated uniformly at random in the range [0.2, 0.8]. The noise probabilities are set to $\nu = 0.01$. For all algorithms, the true structure of the network was provided and only the parameters were left to be estimated. With insufficient data, method-of-moments can estimate parameters outside of the range [0,1]. Any invalid parameters are clipped to lie within $[10^{-6}, 1 - 10^{-6}]$. Since the variational algorithm can become stuck at local maxima, it was seeded with 64 random seeds for each random network and the run that has the best variational lower bound on the likelihood was reported.

Figure 3 shows the L1 error in parameters and run times of the algorithms as a function of the number of samples, averaged over the 64 different networks. Error bars show standard deviation from the mean. The timing test was run on a single machine. Variational EM was run using the authors' C++ implementation of the algorithm[2] and Algorithm 1 was run using a Python implementation. In the large data setting, the method-of-moments algorithm is much faster than variational EM because it only has to iterate through the data once to form empirical estimates of the triplet moments. The variational method requires a pass through the data for every iteration.

In nearly all of the runs, variational EM converges to a set of parameters that effectively assign the children b and c in the network (Figure 1) to one of the two parents A or B by setting the failure probabilities of the other parent to very close to 1. Thus, even though it was provided with the correct structure, the variational EM algorithm effectively pruned out some edges from the network. This bias of the variational EM algorithm towards sparse networks was already noted in Šingliar & Hauskrecht (2006) and appears to be a significant detriment to recovery of the true network parameters.

In addition to the variational EM algorithm, we also show results for EM using exact inference, which is feasible for this simple structure. Exact EM was tested on 16 networks with random parameters and used 4 random initializations, with the run having the best likelihood being reported. These results serve two purposes. First, we want to understand whether the failure of variational EM is due to the error introduced by mean-field inference approximation or due to the fact that EM only reaches a local maxima of the likelihood. The fact that exact EM significantly outperforms variational EM suggests that the problem is with the variational inference. The second purpose is to compare the sample complexity of our method-of-moments approach with a maximum-likelihood based approach. On this small network, the sample complexity of the two approaches appears to be comparable. We emphasize that the exact EM method would be infeasible to run on any reasonably sized network due to the intractability of exact inference in these models.

4.2 Synthetic Data from aQMR-DT

We use the Anonymized QMR Knowledge Base[3] which has the same structure as the true network, but the names of the variables have been obscured and the parameters perturbed. To generate the synthetic data, we transform the parameters of the anonymized knowledge base to parameters of a noisy-or Bayesian network using the procedure described in Morris (2001). The disease priors (not given in aQMR-DT) were sampled according to a Zipf law with exponentially more low probability diseases than high probability diseases.

Using Algorithm 1 extended to take advantage of singly-coupled pairs (as described in Section 3.3), we find a schedule with depth 3 that learns all but a single highly connected subnetwork of QMR-DT. This troublesome subnetwork has two parents, each with 61 children, that overlap on all but one child each (similar to Figure 2 but 60 overlapping children instead of 3). It cannot be learned fully using the main algorithm, though it can be learned with the clean up procedure described in Section 3.3.

The pairs method is very useful for decreasing the maximum depth of the network. Figure 4 (right) compares the depths of parameters learned only with the triplet method to those learned using triplets and pairs combined. Using only triplets eventually learns all of

[2]We thank the authors of Šingliar & Hauskrecht (2006) for kindly providing their implementation.

[3]The QMR Knowledge Base is provided by University of Pittsburgh through the efforts of Frances Connell, Randolph A. Miller, and Gregory F. Cooper.

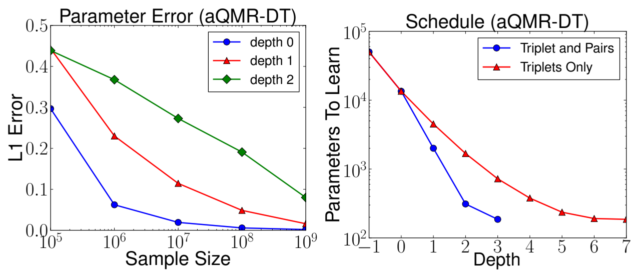


Figure 4: (Left) Mean parameter error as a function of sample size for the failure parameters learned at different depths on the QMR-DT network. Only a small number of failure parameters are learned at depth 3 so it is not included due to its high variance. (Right) Number of parameters (in log scale) left to learn after learning all of the parameters at a given depth, using a schedule that uses both triplets and pairs, compared to a schedule that only uses triplets. At the outset of the algorithm (depth=-1), all of the parameters remain to be learned. The remaining parameters belong to a single subnetwork in the QMR-DT graph that we can learn with the clean up step.

the same parameters as using both triplets and pairs, but requires more adjustment steps.

Figure 4 (left) shows the average L1 error for parameters learned as a function of the depth they were learned at. As expected the error compounds with depth, but with sufficiently large samples, all of the errors tend toward zero. Additionally, as shown in Figure 4 (right), the vast majority of the parameters are learned at depth 0 and 1.

Timings were reported using an AMD-based Dell R815 machine with 64 cores and 256GB RAM. First, a valid schedule to learn all of the parameters of the aQMR-DT network (except the subnetwork described above) was found using Algorithm 1 extended to use pairs. Finding a schedule took 4.5 hours using 32 processors in parallel. Once the schedule is determined, the learning procedure only requires sufficient statistics in the form of the joint distributions of the triplets and pairs and single variables present in the schedule (36,506 triplets, 7,682 pairs and 4,013 singles). The network was sampled and sufficient statistics were computed from each sample. Updating the sufficient statistics took approximately $2.5 \cdot 10^{-4}$ seconds per sample and can be trivially parallelized. Solving for the network parameters using the sufficient statistics takes under 3 minutes with no parallelization at all.

5 Discussion

We presented a method-of-moments approach to learning the parameters of bipartite noisy-or Bayesian networks of known structure and sufficient sparsity, using unlabeled training data that only needs to observe the bottom layer's variables. The method is fast, has theoretical guarantees, and compares favorably to existing variational methods of parameter learning. We show that using this method we can learn almost all of the parameters of the QMR-DT Bayesian network and provide local identifiability results and a method that suggests the remaining parameters can be estimated efficiently as well.

The main algorithm presented in this paper uses third-order moments, but only recovers parameters of a bipartite noisy-or network for a restricted family of network structures. The clean up algorithm can recover all locally identifiable network structures, including fully connected networks, but requires grid searches for parameters that can be exponential in the number of parents. This leaves open the question of whether there are efficient algorithms for recovering a more expansive family of network structures than those covered by the main algorithm.

Provably learning the *structure* of the noisy-or network as well as its parameters from data is more difficult because of identifiability problems. For example, one can show that third-order moments are insufficient for determining the number of hidden variables. We consider this an open problem for further work. Also, in most real-world applications involving expert systems for diagnosis, the hidden variables are not marginally independent (e.g., having diabetes increases the risk of hypertension). It is possible that the techniques described here can be extended to allow for dependencies between the hidden variables.

Another important direction is to attempt to generalize the learning algorithms beyond noisy-or networks of binary variables. The noisy-or distribution is special because adding parents can only *decrease* the negative moments (Eq. 3), and its factorization allows for the effect of individual parents to be isolated. Moreover, since the noisy-or parameterization has a single parameter per hidden variable and observed variable, it is possible to learn part of the model and then hope to adjust the remaining moments (a more general distribution with the same property is the logistic function). New techniques will likely need to be developed to enable learning of arbitrary discrete-valued Bayesian networks with hidden values.

Acknowledgements

We thank Sanjeev Arora, Rong Ge, and Ankur Moitra for early discussions on this work. Research supported in part by a Google Faculty Research Award, CIMIT award 12-1262, grant UL1 TR000038 from NCATS, and by an NSERC Postgraduate Scholarship.

References

Allman, Elizabeth S, Matias, Catherine, & Rhodes, John A. 2009. Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, **37**(6A), 3099–3132.

Anandkumar, A., Hsu, D., & Kakade, S. 2012a. A method of moments for mixture models and hidden Markov models. *In: COLT*.

Anandkumar, Anima, Foster, Dean, Hsu, Daniel, Kakade, Sham, & Liu, Yi-Kai. 2012b. A spectral algorithm for latent Dirichlet allocation. *Pages 926–934 of: Advances in Neural Information Processing Systems 25*.

Anandkumar, Animashree, Hsu, Daniel, Javanmard, Adel, & Kakade, Sham M. 2012c. Learning Linear Bayesian Networks with Latent Variables. *arXiv preprint arXiv:1209.5350*.

Anandkumar, Animashree, Hsu, Daniel, & Kakade, Sham M. 2012d. A method of moments for mixture models and hidden Markov models. *arXiv preprint arXiv:1203.0683*.

Berge, JosM.F. 1991. Kruskal's polynomial for $2 \times 2 \times 2$ arrays and a generalization to $2 \times n \times n$ arrays. *Psychometrika*, **56**, 631–636.

Chang, Joseph T. 1996. Full reconstruction of Markov models on evolutionary trees: identifiability and consistency. *Mathematical biosciences*, **137**(1), 51–73.

Cooper, Gregory F. 1987. *Probabilistic Inference Using Belief Networks Is NP-Hard*. Technical Report BMIR-1987-0195. Medical Computer Science Group, Stanford University.

Heckerman, David E. 1990. *A tractable inference algorithm for diagnosing multiple diseases*. Knowledge Systems Laboratory, Stanford University.

Hsu, D., Kakade, S. M., & Liang, P. 2012. Identifiability and Unmixing of Latent Parse Trees. *In: Advances in Neural Information Processing Systems (NIPS)*.

Jaakkola, Tommi S, & Jordan, Michael I. 1999. Variational Probabilistic Inference and the QMR-DT Network. *Journal of Artificial Intelligence Research*, **10**, 291–322.

Kearns, Michael, & Mansour, Yishay. 1998. Exact inference of hidden structure from sample data in noisy-OR networks. *Pages 304–310 of: Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc.

Miller, Randolph A., Pople, Harry E., & Myers, Jack D. 1982. Internist-I, an Experimental Computer-Based Diagnostic Consultant for General Internal Medicine. *New England Journal of Medicine*, **307**(8), 468–476.

Miller, Randolph A., McNeil, Melissa A., Challinor, Sue M., Fred E. Masarie, Jr., & Myers, Jack D. 1986. The INTERNIST-1/QUICK MEDICAL REFERENCE project – Status report. *West J Med*, **145**(Dec), 816–822.

Morris, Quaid. 2001. Anonymised QMR KB to aQMR-DT.

Mossel, Elchanan, & Roch, Sébastien. 2005. Learning nonsingular phylogenies and hidden Markov models. *Pages 366–375 of: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*. ACM.

Ng, Andrew Y, & Jordan, Michael I. 2000. Approximate inference algorithms for two-layer Bayesian networks. *Advances in neural information processing systems*, **12**.

Shwe, Michael A, Middleton, B, Heckerman, DE, Henrion, M, Horvitz, EJ, Lehmann, HP, & Cooper, GF. 1991. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. *Meth. Inform. Med*, **30**, 241–255.

Šingliar, Tomáš, & Hauskrecht, Miloš. 2006. Noisy-or component analysis and its application to link analysis. *The Journal of Machine Learning Research*, **7**, 2189–2213.

Gaussian Processes for Big Data

James Hensman[*]
Dept. Computer Science
The University of Sheffield
Sheffield, UK

Nicolò Fusi[*]
Dept. Computer Science
The University of Sheffield
Sheffield, UK

Neil D. Lawrence[*]
Dept. Computer Science
The University of Sheffield
Sheffield, UK

Abstract

We introduce stochastic variational inference for Gaussian process models. This enables the application of Gaussian process (GP) models to data sets containing millions of data points. We show how GPs can be variationally decomposed to depend on a set of globally relevant inducing variables which factorize the model in the necessary manner to perform variational inference. Our approach is readily extended to models with non-Gaussian likelihoods and latent variable models based around Gaussian processes. We demonstrate the approach on a simple toy problem and two real world data sets.

1 Introduction

Gaussian processes [GPs, Rasmussen and Williams, 2006] are perhaps the dominant approach for inference on functions. They underpin a range of algorithms for regression, classification and unsupervised learning. Unfortunately, when applying a Gaussian process to a data set of size n exact inference has complexity $\mathcal{O}(n^3)$ with storage demands of $\mathcal{O}(n^2)$. This hinders the application of these models for many domains. In particular, large spatiotemporal data sets, video, large social network data (e.g. from Facebook), population scale medical data sets, models that correlate across multiple outputs or tasks (for these models complexity is $\mathcal{O}(n^3p^3)$ and storage is $\mathcal{O}(n^2p^2)$ where p is the number of outputs or tasks). Collectively we can think of these applications as belonging to the domain of 'big data'.

Traditionally in Gaussian process a large data set is one that contains over a few thousand data points. Even to accommodate these data sets, various approximate techniques are required. One approach is to partition the data set into separate groups [e.g. Snelson and Ghahramani, 2007, Urtasun and Darrell, 2008]. An alternative is to build a low rank approximation to the covariance matrix based around 'inducing variables' [see e.g. Csató and Opper, 2002, Seeger et al., 2003, Quiñonero Candela and Rasmussen, 2005, Titsias, 2009]. These approaches lead to a computational complexity of $\mathcal{O}(nm^2)$ and storage demands of $\mathcal{O}(nm)$ where m is a user selected parameter governing the number of inducing variables. However, even these reduced storage are prohibitive for big data, where n can be many millions or billions. For parametric models, stochastic gradient descent is often applied to resolve this storage issue, but in the GP domain, it hasn't been clear how this should be performed. In this paper we show how recent advances in variational inference [Hensman et al., 2012, Hoffman et al., 2012] can be combined with the idea of inducing variables to develop a practical algorithm for fitting GPs using stochastic variational inference (SVI).

2 Sparse GPs Revisited

We start with a succinct rederivation of the variational approach to inducing variables of Titsias [2009]. This allows us to introduce notation and derive expressions which allow for the formulation of a SVI algorithm.

Consider a data vector[1] $\mathbf{y}$, where each entry y_i is a noisy observation of the function $f(\mathbf{x}_i)$, for all the points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$. We consider the noise to be independent Gaussian with precision β. Introducing a Gaussian process prior over $f(\cdot)$, let the vector $\mathbf{f}$ contain values of the function at the points $\mathbf{X}$. We shall also introduce a set of *inducing variables*: let the vector $\mathbf{u}$ contain values of the function f at the points $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^m$ which live in the same space as $\mathbf{X}$. Us-

[*]Also at Sheffield Institute for Translational Neuroscience, SITraN

[1]Our derivation trivially extends to multiple independent output dimensions, but we omit them here for clarity.

ing standard Gaussian process methodologies, we can write

$$p(\mathbf{y}\,|\,\mathbf{f}) = \mathcal{N}\left(\mathbf{y}|\mathbf{f}, \beta^{-1}\mathbf{I}\right),$$
$$p(\mathbf{f}\,|\,\mathbf{u}) = \mathcal{N}\left(\mathbf{f}|\mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{u}, \widetilde{\mathbf{K}}\right),$$
$$p(\mathbf{u}) = \mathcal{N}\left(\mathbf{u}|\mathbf{0}, \mathbf{K}_{mm}\right),$$

where $\mathbf{K}_{mm}$ is the covariance function evaluated between all the inducing points and $\mathbf{K}_{nm}$ is the covariance function between all inducing points and training points and we have defined with $\widetilde{\mathbf{K}} = \mathbf{K}_{nn} - \mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{mn}$.

We first apply Jensen's inequality on the conditional probability $p(\mathbf{y}\,|\,\mathbf{u})$:

$$\log p(\mathbf{y}\,|\,\mathbf{u}) = \log \left\langle p(\mathbf{y}\,|\,\mathbf{f})\right\rangle_{p(\mathbf{f}\,|\,\mathbf{u})}$$
$$\geq \left\langle \log p(\mathbf{y}\,|\,\mathbf{f})\right\rangle_{p(\mathbf{f}\,|\,\mathbf{u})} \triangleq \mathcal{L}_1. \qquad (1)$$

where $\langle\cdot\rangle_{p(x)}$ denotes an expectation under $p(x)$. For Gaussian noise taking the expectation inside the log is tractable, but it results in an expression containing $\mathbf{K}_{nn}^{-1}$, which has a computational complexity of $\mathcal{O}(n^3)$. Bringing the expectation outside the log gives a lower bound, $\mathcal{L}_1$, which can be computed with has complexity $\mathcal{O}(m^3)$. Further, when $p(\mathbf{y}|\mathbf{f})$ factorises across the data,

$$p(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{n} p(y_i|f_i),$$

then this lower bound can be shown to be separable across $\mathbf{y}$ giving

$$\exp(\mathcal{L}_1) = \prod_{i=1}^{n} \mathcal{N}\left(y_i|\mu_i, \beta^{-1}\right) \exp\left(-\frac{1}{2}\beta\tilde{k}_{i,i}\right) \qquad (2)$$

where $\boldsymbol{\mu} = \mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{u}$ and $\tilde{k}_{i,i}$ is the ith diagonal element of $\widetilde{\mathbf{K}}$. Note that the difference between our bound and the original log likelihood is given by the Kullback Leibler (KL) divergence between the posterior over the mapping function given the data and the inducing variables and the posterior of the mapping function given the inducing variables only,

$$\mathrm{KL}\left(p(\mathbf{f}|\mathbf{u})\,\|\,p(\mathbf{f}|\mathbf{u},\mathbf{y})\right).$$

This KL divergence is minimized when there are $m = n$ inducing variables and they are placed at the training data locations. This means that $\mathbf{u} = \mathbf{f}$, $\mathbf{K}_{mm}$=$\mathbf{K}_{nm}$=$\mathbf{K}_{nn}$ meaning that $\widetilde{\mathbf{K}}$=$\mathbf{0}$. In this case we recover $\exp(\mathcal{L}_1) = p(\mathbf{y}|\mathbf{f})$ and the bound becomes equality because $p(\mathbf{f}|\mathbf{u})$ is degenerate. However, since $m = n$ and that there would be no computational or storage advantage from the representation. When $m < n$ the bound can be maximised with respect to $\mathbf{Z}$ (which are variational parameters). This minimises the KL divergence and ensures that $\mathbf{Z}$ are distributed amongst the training data $\mathbf{X}$ such that all $\tilde{k}_{i,i}$ are small. In practice this means that the expectations in (1) are only taken across a narrow domain ($\tilde{k}_{i,i}$ is the marginal variance of $p(f_i|\mathbf{u})$), keeping Jensen's bound tight.

Before deriving the expressions for stochastic variational inference using $\mathcal{L}_1$, we recover the bound of Titsias [2009] by marginalising the inducing variables,

$$\log p(\mathbf{y}\,|\,\mathbf{X}) = \log \int p(\mathbf{y}\,|\,\mathbf{u})p(\mathbf{u})\,\mathrm{d}\mathbf{u}$$
$$\geq \log \int \exp\left\{\mathcal{L}_1\right\} p(\mathbf{u})\,\mathrm{d}\mathbf{u} \triangleq \mathcal{L}_2, \qquad (3)$$

which with some linear algebraic manipulation leads to

$$\mathcal{L}_2 = \log \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{mn} + \beta^{-1}\mathbf{I}\right) - \frac{1}{2}\beta \mathrm{tr}\left(\widetilde{\mathbf{K}}\right),$$

matching the result of Titsias, with the implicit approximating distribution $q(\mathbf{u})$ having precision

$$\boldsymbol{\Lambda} = \beta\mathbf{K}_{mm}^{-1}\mathbf{K}_{mn}\mathbf{K}_{nm}\mathbf{K}_{mm}^{-1} + \mathbf{K}_{mm}^{-1}$$

and mean

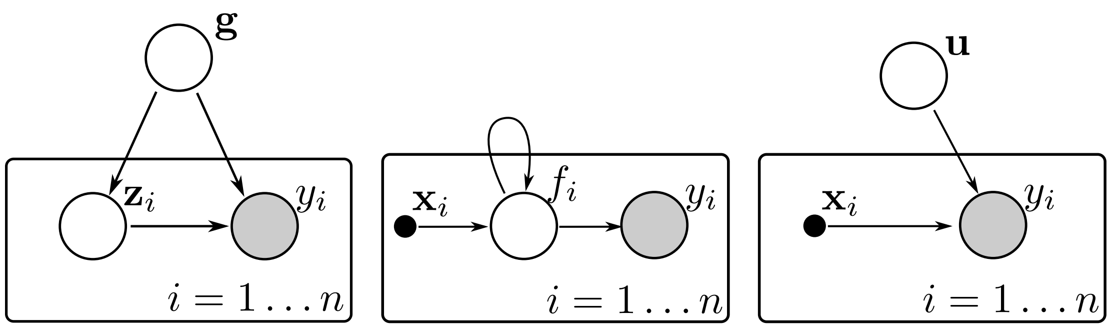
$$\hat{\mathbf{u}} = \beta\boldsymbol{\Lambda}^{-1}\mathbf{K}_{mm}^{-1}\mathbf{K}_{mn}\mathbf{y}.$$

3 SVI for GPs

One of the novelties of the Titsias bound was that, rather than explicitly representing a variational distribution for $q(\mathbf{u})$, these variables are 'collapsed' [Hensman et al., 2012]. However, for stochastic variational inference to work on Gaussian processes, it turns out we need to maintain an explicit representation of these inducing variables.

Stochastic variational inference (SVI) allows variational inference for very large data sets, but it can only be applied to probabilistic models which have a set of *global* variables, and which factorise in the observations and latent variables as Figure 1(a). Gaussian Processes do not have global variables and exhibit no such factorisation (Figure 1(b)). By introducing inducing variables $\mathbf{u}$, we have an appropriate model for SVI (Figure 1(c)). Unfortunately, marginalising $\mathbf{u}$ re-introduces dependencies between the observations, and eliminates the global parameters. In the following, we derive a lower bound on $\mathcal{L}_2$ which includes an explicit variational distribution $q(\mathbf{u})$, enabling SVI. We then derive the required natural gradients and discuss how latent variables might be used.

Because there are a fixed number of inducing variables (specified by the user at algorithm design time) we can perform stochastic variational inference, greatly increasing the size of data sets to which we can apply Gaussian processes.



(a) Requirements for SVI (b) Gaussian Process regression (c) Variational GP regression

Figure 1: Graphical models showing (a) the reqired form for a probabilistic model for SVI (reproduced from [Hoffman et al., 2012]), with *global* variables $\mathbf{g}$ and latent variables $\mathbf{z}$. (b) The graphical model corresponding to Gaussian process regression, where connectivity between the values of the function f_i is denoted by a loop around the plate. (c) The graphical model corresponding to the sparse GP model, with inducing variables $\mathbf{u}$ working as global variables, and the term $\mathcal{L}_1$ acting as $\log p(y_i \,|\, \mathbf{u}, \mathbf{x}_i)$. Marginalisation of $\mathbf{u}$ leads to the variational DTC formulation, introducing dependencies between the observations.

3.1 Global Variables

To apply stochastic variational inference to a Gaussian process model, we must have a set of global variables. The variables $\mathbf{u}$ will perform this role, and we introduce a variational distribution $q(\mathbf{u})$, and use it to lower bound the quantity $p(\mathbf{y} \,|\, \mathbf{X})$.

$$\log p(\mathbf{y} \,|\, \mathbf{X}) \geq \langle \mathcal{L}_1 + \log p(\mathbf{u}) - \log q(\mathbf{u}) \rangle_{q(\mathbf{u})} \triangleq \mathcal{L}_3.$$

From the above we know that the optimal distribution is Gaussian, and we parametrise it as $q(\mathbf{u}) = \mathcal{N}(\mathbf{u} \,|\, \mathbf{m}, \mathbf{S})$. The bound $\mathcal{L}_3$ becomes

$$\begin{aligned}\mathcal{L}_3 = \sum_{i=1}^{n} \Big\{ &\log \mathcal{N}\left(y_i | \mathbf{k}_i^\top \mathbf{K}_{mm}^{-1} \mathbf{m}, \beta^{-1}\right) \\ &- \frac{1}{2}\beta \widetilde{k}_{i,i} - \frac{1}{2}\mathrm{tr}\left(\mathbf{S}\boldsymbol{\Lambda}_i\right) \Big\} \\ &- \mathrm{KL}\left(q(\mathbf{u}) \,\|\, p(\mathbf{u})\right) \end{aligned} \tag{4}$$

with $\mathbf{k}_i$ being a vector of the i^{th} column of $\mathbf{K}_{mn}$ and $\boldsymbol{\Lambda}_i = \beta \mathbf{K}_{mm}^{-1} \mathbf{k}_i \mathbf{k}_i^\top \mathbf{K}_{mm}^{-1}$. The gradients of $\mathcal{L}_3$ with respect to the parameters of $q(\mathbf{u})$ are

$$\begin{aligned}\frac{\partial \mathcal{L}_3}{\partial \mathbf{m}} &= \beta \mathbf{K}_{mm}^{-1} \mathbf{K}_{mn} \mathbf{y} - \boldsymbol{\Lambda}\mathbf{m}, \\ \frac{\partial \mathcal{L}_3}{\partial \mathbf{S}} &= \frac{1}{2}\mathbf{S}^{-1} - \frac{1}{2}\boldsymbol{\Lambda}. \end{aligned} \tag{5}$$

Setting the derivatives to zero recovers the optimal solution found in the previous section, namely $\mathbf{S}{=}\boldsymbol{\Lambda}^{-1}$, $\mathbf{m}{=}\hat{\mathbf{u}}$. It follows that $\mathcal{L}_2 \geq \mathcal{L}_3$, with equality at this unique maximum.

The key propery of $\mathcal{L}_3$ is that is can be written as a sum of n terms, each corresponding to one input-output pair $\{\mathbf{x}_i, y_i\}$: we have induced the necessary factorisation to perform stochastic gradient methods on the distribution $q(\mathbf{u})$.

3.2 Natural Gradients

Stochastic variational inference works by taking steps in the direction of the approximate *natural* gradient $\widetilde{\mathbf{g}}(\boldsymbol{\theta})$, which is given by the usual gradient re-scaled by the inverse Fisher information: $\widetilde{\mathbf{g}}(\boldsymbol{\theta}){=}G(\boldsymbol{\theta})^{-1}\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}}$. To work with the natural gradients of the distribution $q(\mathbf{u})$, we first recall the canonical and expectation parameters

$$\boldsymbol{\theta}_1{=}\mathbf{S}^{-1}\mathbf{m}, \quad \boldsymbol{\theta}_2{=}{-}\frac{1}{2}\mathbf{S}^{-1}$$

and

$$\boldsymbol{\eta}_1{=}\mathbf{m}, \quad \boldsymbol{\eta}_2{=}\mathbf{m}\mathbf{m}^\top{+}\mathbf{S}.$$

In the exponential family, properties of the Fisher information reveal the following simplification of the natural gradient [Hensman et al., 2012],

$$\widetilde{\mathbf{g}}(\boldsymbol{\theta}) = G(\boldsymbol{\theta})^{-1}\frac{\partial \mathcal{L}_3}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{L}_3}{\partial \boldsymbol{\eta}}. \tag{6}$$

A step of length ℓ in the natural gradient direction, using $\boldsymbol{\theta}_{(t+1)} = \boldsymbol{\theta}_{(t)} + \ell \frac{\mathrm{d}\mathcal{L}_3}{\mathrm{d}\boldsymbol{\eta}}$, yields

$$\begin{aligned}\boldsymbol{\theta}_{2(t+1)} &= -\frac{1}{2}\mathbf{S}_{(t+1)}^{-1} \\ &= -\frac{1}{2}\mathbf{S}_{(t)}^{-1} + \ell\left(-\frac{1}{2}\boldsymbol{\Lambda} + \frac{1}{2}\mathbf{S}_{(t)}^{-1}\right), \\ \boldsymbol{\theta}_{1(t+1)} &= \mathbf{S}_{(t+1)}^{-1}\mathbf{m}_{(t+1)} \\ &= \mathbf{S}_{(t)}^{-1}\mathbf{m}_{(t)} + \ell\left(\beta\mathbf{K}_{mm}^{-1}\mathbf{K}_{mn}\mathbf{y} - \mathbf{S}_{(t)}^{-1}\mathbf{m}_{(t)}\right),\end{aligned}$$

and taking a step of unit length then recovers the same solution as above by either (3) or (5). This confirms the result discussed in Hensman et al. [2012], Hoffman et al. [2012], that taking this unit step is the same as performing a VB update. We can now obtain stochastic approximations to the natural gradient by considering the data either individually or in mini-batches.

We note the convenient result that the natural gradient for $\boldsymbol{\theta}_2$ is positive definite (note $\boldsymbol{\Lambda} = \mathbf{K}_{mm}^{-1} + \sum_i \boldsymbol{\Lambda}_i$). This means that taking a step in that direction always leads to a positive definite matrix, and our implementation need not parameterise $\mathbf{S}$ in any way so as to ensure positive-definiteness, *cf.* standard gradient approaches on covariance matrices.

To optimise the kernel hyper-parameters and noise precision β, we take derivatives of the bound $\mathcal{L}_3$ and perform standard stochastic gradient descent alongside the variational parameters. An illustration is presented in Figure 2.

3.3 Latent Variables

The above derivations enable online learning for Gaussian process *regression* using SVI. Several GP based models involve inference of $\mathbf{X}$, such as the GP latent variable model [Lawrence, 2005, Titsias and Lawrence, 2010] and its extensions [e.g. Damianou et al., 2011, 2012].

To perform stochastic variational inference with latent variables, we require a factorisation as illustrated by Figure 1(a): this factorisation is provided by (4). To get a model like the Bayesian GPLVM, we need a lower bound on $\log p(\mathbf{y})$. In Titsias and Lawrence [2010] this was achieved through approximate marginalisation of $\mathcal{L}_2$, w.r.t. $\mathbf{X}$, which leads to an expression depending only on the parameters of $q(\mathbf{X})$. However this formulation scales poorly, and the variables of the optimisation are closely connected due to the marginalisation of $\mathbf{u}$. The above enables a lower bound to which SVI is immediately applicable:

$$\begin{aligned}\log p(\mathbf{y}) &= \log \int p(\mathbf{y}\,|\,\mathbf{X}) p(\mathbf{X})\,\mathrm{d}\mathbf{X} \\ &\geq \int q(\mathbf{X}) \big\{ \mathcal{L}_3 + \log p(\mathbf{X}) - \log q(\mathbf{X}) \big\}\,\mathrm{d}\mathbf{X}.\end{aligned}$$

It is straightforward to introduce p output dimensions for the data $\mathbf{Y}$, and following Titsias and Lawrence [2010], we use a factorising normal distribution $q(\mathbf{X}) = \prod_{i=1}^{n} q(\mathbf{x}_i)$. The relevant expectations of $\mathcal{L}_3$ are tractable for various choices of covariance function.

To perform SVI in this model, we now alternate between selecting a minibatch of data, and optimisting the relevant variables of $q(\mathbf{X})$ with $q(\mathbf{u})$ fixed, and updating $q(\mathbf{u})$ using the approximate natural gradient. We note that the form of (4) means that each of the latent variable distributions may be updated individually, enabling parallelisation across the minibatch.

3.4 Non-Gaussian likelihoods

Another advantage of the factorisation of (4) is that it enables a simple routine for inference with non-Gaussian likelihoods. The usual procedure for fitting GPs with non-Gaussian likelihoods is to approximate the likelihood using either a local variational lower bound [Gibbs and MacKay, 2000], or by expectation propagation [Kuss and Rasmussen, 2005]. These approximations to the likelihood are required because of the connections between the variables $\mathbf{f}$.

In $\mathcal{L}_3$, the bound factorises in such a way that some non-Gaussian likelihoods may be marginalised *exactly*, given the existing approximations. To see this, consider that we are presented not with the vector $\mathbf{y}$, but by a binary vector $\mathbf{t}$ with $t_i \in \{0, 1\}$, and the likelihood $p(\mathbf{t}\,|\,\mathbf{y}) = \prod_{i=1}^{n} \sigma(y_i)^{t_i} (1-\sigma(y_i))^{(1-t_i)}$, as in the case of classification. We can bound the marginal likelihood using $p(\mathbf{t}\,|\,\mathbf{X}) \geq \int p(\mathbf{t}\,|\,\mathbf{y}) \exp\{\mathcal{L}_3\}\,\mathrm{d}\mathbf{y}$ which involves n independent one dimensional integrals due to the factorising nature of $\mathcal{L}_3$. For the probit likelihood each of these integrals is tractable.

This kind of approximation, where the likelihood is integrated exactly is amenable to SVI in the same manner as the regression case above through computation of the natural gradient.

4 Experiments

4.1 Toy Data

To demonstrate our algorithm we begin with two simple toy datasets based on sinusoidal functions. In the first experiment we show how the approximation converges towards the true solution as mini-batches are included. Figure 2 shows the nature of our approximation: the variational approximation to the inducing function variables is shown.

The second toy problem (Figure 3) illustrates the convergence of the algorithm on a two dimensional problem, again based on sinusoids. Here, we start with a random initialisation for $q(\mathbf{u})$, and the model converges after 2000 iterations. We found empirically that holding the covariance parameters fixed for the first epoch results in more reliable convergence, as can be seen in Figure 4

4.2 UK Apartment Price Data

Our first large scale Gaussian process models the changing cost of apartments in the UK. We downloaded the monthly price paid data for the period February to October 2012 from `http://data.gov.uk/dataset/`

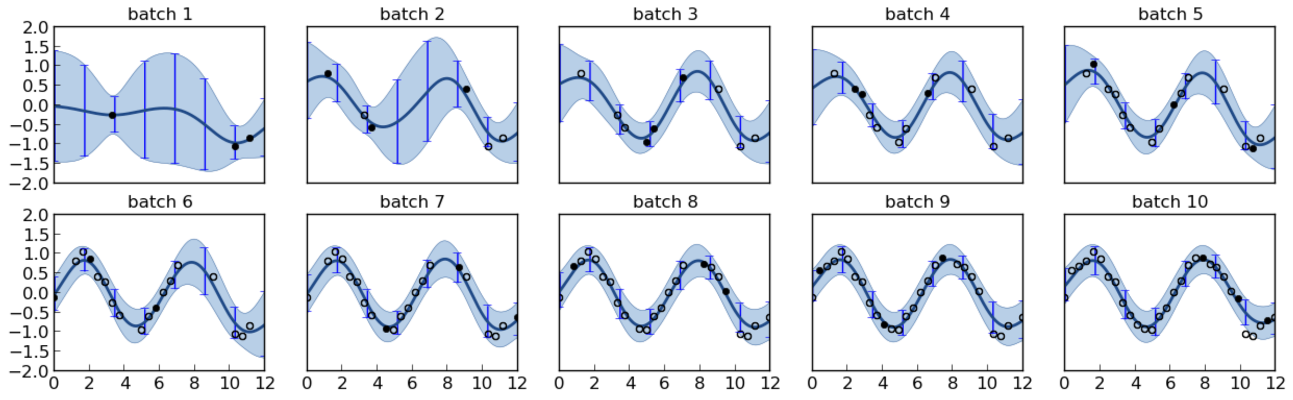
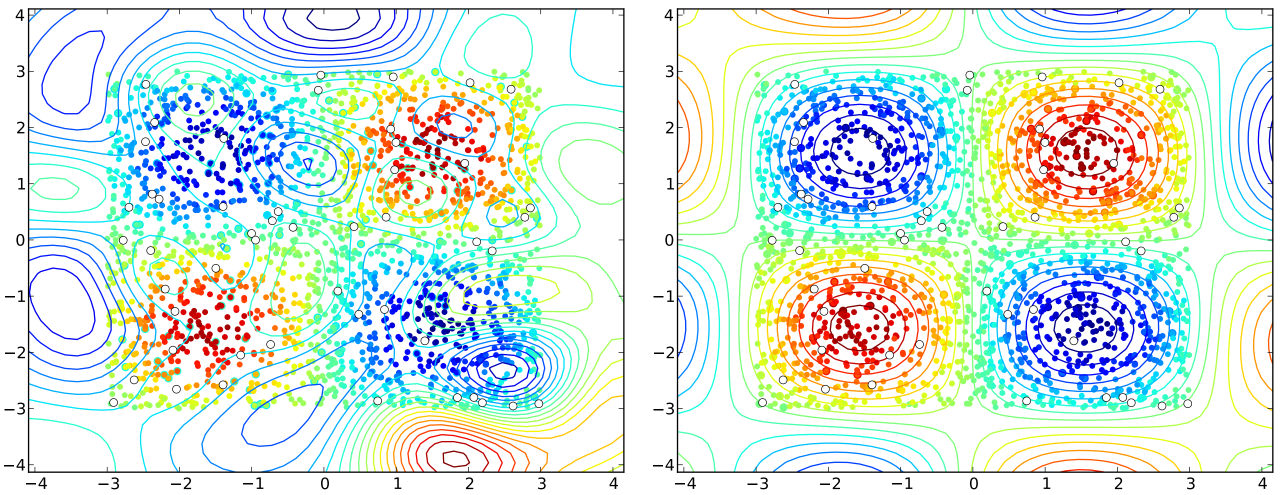


Figure 2: Stochastic variational inference on a trivial GP regression problem. Each pane shows the posterior of the GP after a batch of data, marked as solid points. Previoulsy seen (and discarded) data are marked as empty points, the distribution $q(\mathbf{u})$ is represented by vertical errorbars.



(a) Initial condition

(b) Condition at 2000 iterations

Figure 3: A two dimensional toy demo, showing the initial condition and final condition of the model. Data are marked as colored points, and the model's prediction is shown as (similarly colored) contour lines. The positions of the inducing variables are marked as empty circles.

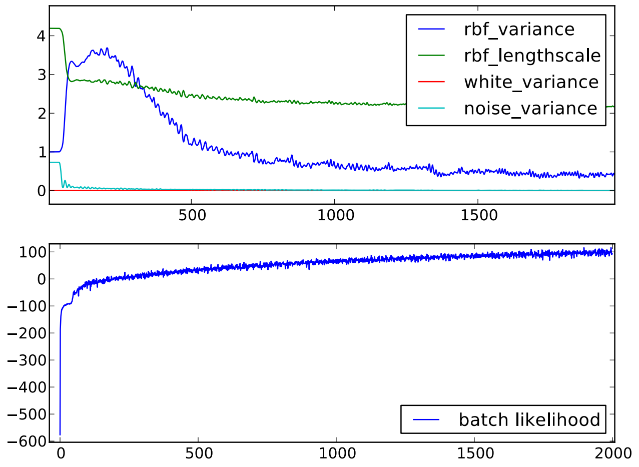


Figure 4: Convergence of the SVIGP algorithm on the two dimensional toy data

`land-registry-monthly-price-paid-data/`, which covers England and Wales, and filtered for apartments. This resulted in a data set with 75,000 entries, which we cross referenced against a postcode database to get lattitude and longitude, on which we regressed the normalised logarithm of the apartment prices.

Randomly selecting 10,000 data as a test set, we build a GP as described with a covariance function $k(\cdot, \cdot)$ consisting of four parts: two squared exponential covariances, initialised with different length scales were used to account for national and regional variations in property prices, a constant (or 'bias') term allowed for non-zero mean data, and a noise variance accounted for variation that could not be modelled using simply latitude and longitude.

We selected 800 inducing input sites using a k-means algorithm, and optimised the parameters of the covariance function alongside the variational parameters. We performed some manual tuning of the learning rates: empirically we found that the step length should be much higher for the variational parameters of $q(\mathbf{u})$ than for the values of the covariance function parameters. We used 0.01 and 1×10^{-5}. Also, we included a momentum term for the covariance function parameters (set to 0.9). We tried including momentum terms for the variational parameters, but we found this hindered performance. A large mini-batch size (1000) reduced the stochasticity of the gradient computations. We judged that the algorithm had converged after 750 iterations, as the stochastic estimate of the marginal lower bound on the marginal likelihood failed to increase further.

For comparison to our model, we constructed a series of GPs on subsets of the training data. Splitting the data into sets of 500, 800, 1000 and 1200, we fitted a GP with the same covariance function as our stochastic GP. Parameters of the covariance function were optimised using type-II maximum likelihood for each batch. Table 1 reports the mean squared error in our model's prediction of the held out prices, as well as the same for the random sub-set approach (along with two standard deviations of the inter-sub-set variability).

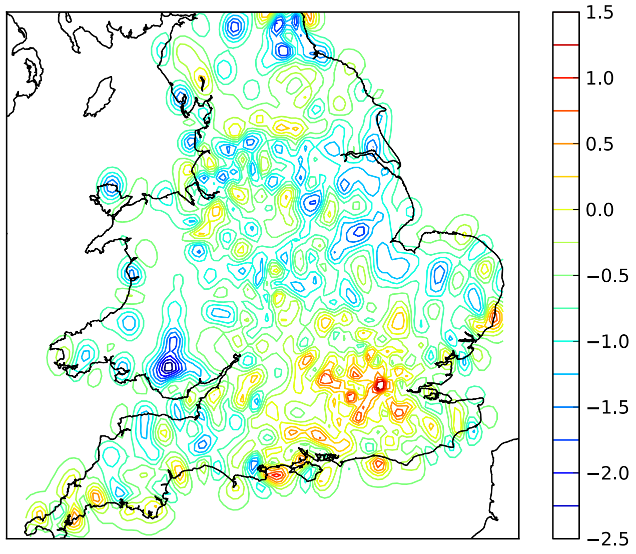


Figure 5: Variability of apartment price (logarithmically!) throughout England and Wales.

Table 1: Mean squared errors in predicting the log-apartment prices across England and Wales by lattitude and longitude

	Mean square Error
SVIGP	**0.426**
Random sub-set (N=500)	0.522 +/- 0.018
Random sub-set (N=800)	0.510 +/- 0.015
Random sub-set (N=1000)	0.503 +/- 0.011
Random sub-set (N=1200)	0.502 +/- 1.012

4.3 Airline Delays

The second large scale dataset we considered consists of flight arrival and departure times for every commercial flight in the USA from January 2008 to April 2008. This dataset contains extensive information about almost 2 million flights, including the delay (in minutes) in reaching the destination. The average delay of a flight in the first 4 months of 2008 was of 30 minutes. Of course, much better estimates can be given by exploiting the enourmous wealth of data available, but rich models are often overlooked in these cases due

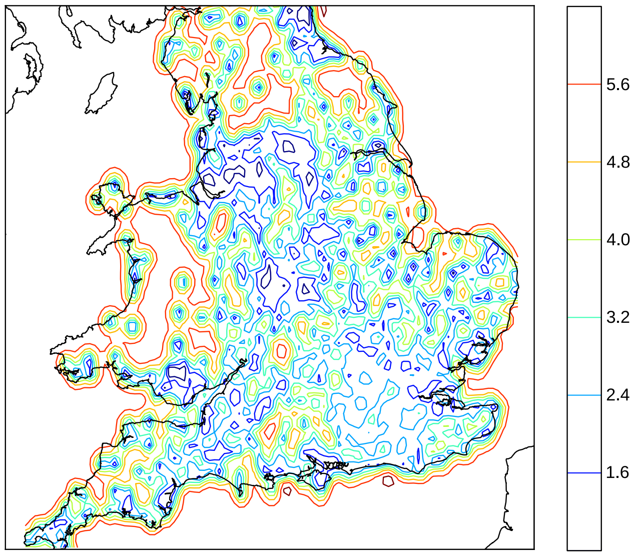


Figure 6: Posterior variance of apartment prices.

to the sheer size of the dataset. We randomly selected 800,000 datapoints [2], using a random subset of 700,000 samples to train the model and 100,000 to test it. We chose to include into our model 8 of the many variables available for this dataset: the age of the aircraft (number of years since deployment), distance that needs to be covered, airtime, departure time, arrival time, day of the week, day of the month and month.

We built a Gaussian process with a squared exponential covariance function with a bias and noise term. In order to discard irrelevant input dimensions, we allowed a separate lengthscale for each input. For our experiments, we used $m = 1000$ inducing inputs and a mini-batch size of 5000. The learning rate for the variational parameters of $q(\mathbf{u})$ was set to 0.01, while the learning rate for the covariance function parameters was set to 1×10^{-5}. We also used a momentum term of 0.9 for the covariance parameters.

For the purpose of comparison, we fitted several GPs with an identical covariance function on subsets of the data. We split the data into sets of 800, 1000 and 1200 samples and optimised the parameters using type-II maximum likelihood. We repeated this procedure 10 times.

The left pane of Figure 7 shows the root mean squared error (RMSE) obtained by fitting GPs on subsets of the data. The right pane of figure 7 shows the RMSE obtained by fitting 10 SVI GPs as a function of the iteration. The individual runs are shown in light gray, while the blue line shows the average RMSE across runs.

[2]Subsampling wasn't technically necessary, but we didn't want to overburden the memory of a shared compute node just before a submission deadline.

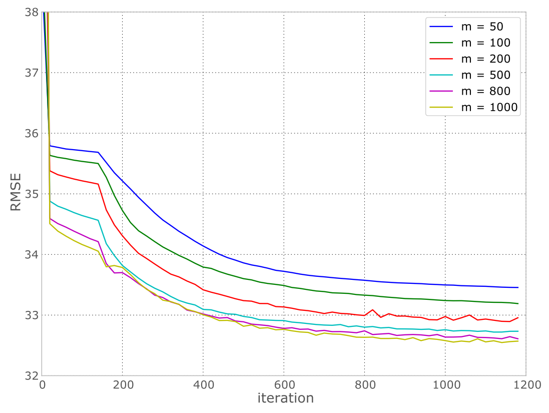


Figure 8: Root mean square errors for models with different numbers of inducing variables.

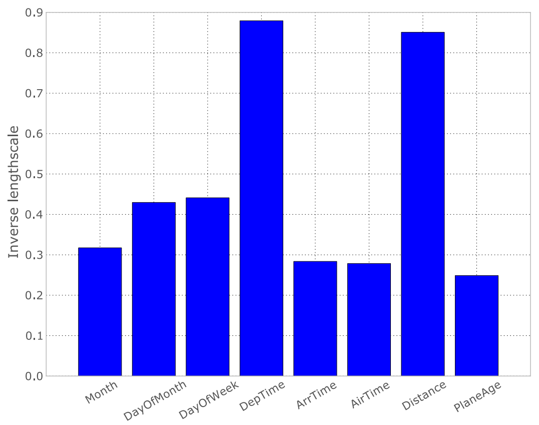


Figure 9: Automatic relevance determination parameters for the features used for predicting flight delays.

One of the main advantages of the approach presented here is that the computational complexity is independent from the number of samples n. This allowed us to use a much larger number of inducing inputs than has traditionally been possible. Conventional sparse GPs have a computational complexity of $\mathcal{O}(nm^2)$, so for large n the typical upper bound for m is between 50 and 100. The impact on the prediction performance is quite significant, as highlighted in Figure 8, where we fit several SVI GPs using different numbers of inducing inputs.

Looking at the inverse lengthscales in Figure 9, it's possible to get a better idea of the relevance of the different features available in this dataset. The most relevant variable turned out to be the time of departure of the flight, closely followed by the distance that needs

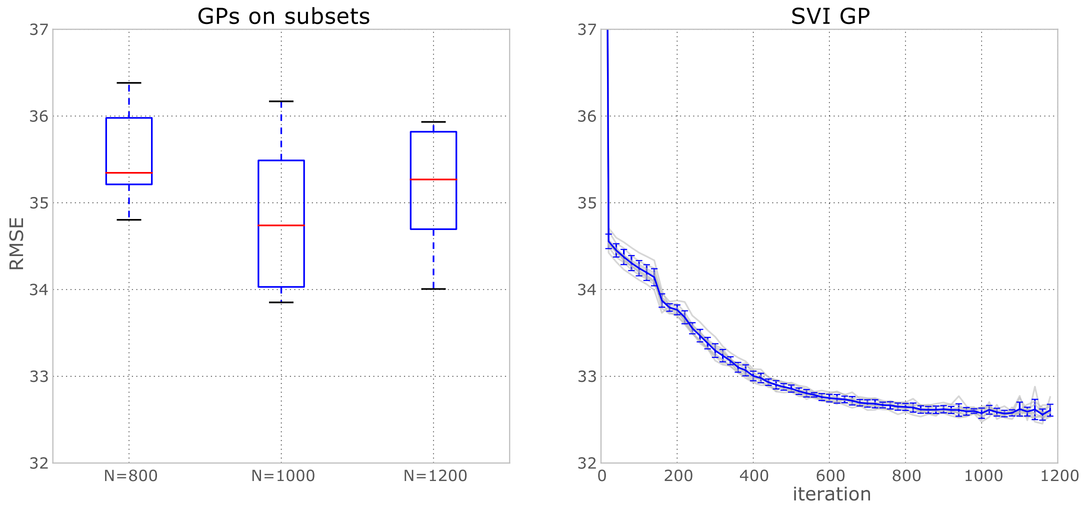


Figure 7: Root mean squared errors in predicting flight delays using information about the flight.

to be covered. Distance and airtime should in theory be correlated, but they have very different relevances. This can be intuitively explained by considering that on longer flights it's easier to make up for delays at departure.

5 Discussion

We have presented a method for inference in Gaussian process models using stochastic variational inference. These expressions allow for the transfer of a multitude of Gaussian process techniques to big data.

We note several interesting results. First, the our derivation disusses the bound on $p(\mathbf{y}\,|\,\mathbf{u})$ in detail, showing that it becomes tight when $\mathbf{Z} = \mathbf{X}$.

Also, we have that there is a unique solution for the parameters of $q(\mathbf{u})$ such that the bound associated with the standard variational sparse GP [Titsias, 2009] is recovered.

Further, since the complexity of our model is now $\mathcal{O}(m^3)$ rather than $\mathcal{O}(nm^2)$, we are free to increase m to much greater values than the sparse GP representation. The effect of this is that we can have much richer models: for a squared exponential covariance function, we have far more basis-functions with which to model the data. In our UK apartment price example, we had no difficulty setting m to 800, much higher than experience tells us is feasible with the sparse GP.

The ability to increase the number of inducing variables and the applicability to unlimited data make our method suitable for multiple output GPs [Álvarez and Lawrence, 2011]. We have also briefly discussed how this framework fits with other Gaussian process based models such as the GPLVM and GP classification. We leave the details of these implementations to future work.

In all our experiments our algorithm was run on a single CPU using the GPy Gaussian process toolkit `https://github.com/SheffieldML/GPy`.

References

Mauricio A. Álvarez and Neil D. Lawrence. Computationally efficient convolved multiple output Gaussian processes. *Journal of Machine Learning Research*, 12:1425–1466, May 2011.

Lehel Csató and Manfred Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

Andreas Damianou, Michalis K. Titsias, and Neil D. Lawrence. Variational Gaussian process dynamical systems. In Peter Bartlett, Fernando Peirrera, Chris Williams, and John Lafferty, editors, *Advances in Neural Information Processing Systems*, volume 24, Cambridge, MA, 2011. MIT Press.

Andreas Damianou, Carl Henrik Ek, Michalis K. Titsias, and Neil D. Lawrence. Manifold relevance determination. In John Langford and Joelle Pineau, editors, *Proceedings of the International Conference in Machine Learning*, volume 29, San Francisco, CA, 2012. Morgan Kauffman. To appear.

Mark N. Gibbs and David J. C. MacKay. Variational Gaussian process classifiers. *IEEE Transactions on Neural Networks*, 11(6):1458–1464, 2000.

James Hensman, Magnus Rattray, and Neil D.

Lawrence. Fast variational inference in the exponential family. *NIPS 2012*, 2012.

Matthew Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *arXiv preprint arXiv:1206.7051*, 2012.

Malte Kuss and Carl Edward Rasmussen. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005.

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 11 2005.

Joaquin Quiñonero Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 0-262-18253-X.

Matthias Seeger, Christopher K. I. Williams, and Neil D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In Christopher M. Bishop and Brendan J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, FL, 3–6 Jan 2003.

Edward Snelson and Zoubin Ghahramani. Local and global sparse Gaussian process approximations. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Workshop on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 21-24 March 2007. Omnipress.

Michalis K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Workshop on Artificial Intelligence and Statistics*, volume 5, pages 567–574, Clearwater Beach, FL, 16-18 April 2009. JMLR W&CP 5.

Michalis K. Titsias and Neil D. Lawrence. Bayesian Gaussian process latent variable model. In Yee Whye Teh and D. Michael Titterington, editors, *Proceedings of the Thirteenth International Workshop on Artificial Intelligence and Statistics*, volume 9, pages 844–851, Chia Laguna Resort, Sardinia, Italy, 13-16 May 2010. JMLR W&CP 9.

Raquel Urtasun and Trevor Darrell. Local probabilistic regression for activity-independent human pose inference. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, 2008.

Inverse Covariance Estimation for High-Dimensional Data in Linear Time and Space: Spectral Methods for Riccati and Sparse Models

Jean Honorio
CSAIL, MIT
Cambridge, MA 02139, USA
jhonorio@csail.mit.edu

Tommi Jaakkola
CSAIL, MIT
Cambridge, MA 02139, USA
tommi@csail.mit.edu

Abstract

We propose maximum likelihood estimation for learning Gaussian graphical models with a Gaussian (ℓ_2^2) prior on the parameters. This is in contrast to the commonly used Laplace (ℓ_1) prior for encouraging sparseness. We show that our optimization problem leads to a Riccati matrix equation, which has a closed form solution. We propose an efficient algorithm that performs a singular value decomposition of the training data. Our algorithm is $\mathcal{O}(NT^2)$-time and $\mathcal{O}(NT)$-space for N variables and T samples. Our method is tailored to high-dimensional problems ($N \gg T$), in which sparseness promoting methods become intractable. Furthermore, instead of obtaining a single solution for a specific regularization parameter, our algorithm finds the whole solution path. We show that the method has logarithmic sample complexity under the spiked covariance model. We also propose sparsification of the dense solution with provable performance guarantees. We provide techniques for using our learnt models, such as removing unimportant variables, computing likelihoods and conditional distributions. Finally, we show promising results in several gene expressions datasets.

1 Introduction

Estimation of large inverse covariance matrices, particularly when the number of variables N is significantly larger than the number of samples T, has attracted increased attention recently. One of the main reasons for this interest is the need for researchers to discover interactions between variables in high dimensional datasets, in areas such as genetics, neuroscience and meteorology. For instance in gene expression datasets, N is in the order of 20 thousands to 2 millions, while T is in the order of few tens to few hundreds. Inverse covariance (precision) matrices are the natural parameterization of Gaussian graphical models.

In this paper, we propose maximum likelihood estimation for learning Gaussian graphical models with a Gaussian (ℓ_2^2) prior on the parameters. This is in contrast to the commonly used Laplace (ℓ_1) prior for encouraging sparseness. We consider the computational aspect of this problem, under the assumption that the number of variables N is significantly larger than the number of samples T, i.e. $N \gg T$.

Our technical contributions in this paper are the following. First, we show that our optimization problem leads to a Riccati matrix equation, which has a closed form solution. Second, we propose an efficient algorithm that performs a singular value decomposition of the sample covariance matrix, which can be performed very efficiently through singular value decomposition of the training data. Third, we show logarithmic sample complexity of the method under the spiked covariance model. Fourth, we propose sparsification of the dense solution with provable performance guarantees. That is, there is a bounded degradation of the Kullback-Leibler divergence and expected log-likelihood. Finally, we provide techniques for using our learnt models, such as removing unimportant variables, computing likelihoods and conditional distributions.

2 Background

In this paper, we use the notation in Table 1.

A *Gaussian graphical model* is a graph in which all random variables are continuous and jointly Gaussian. This model corresponds to the multivariate normal distribution for N variables with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{N \times N}$. Conditional independence in a

Table 1: Notation used in this paper.

Notation	Description
$\mathbf{A} \succeq \mathbf{0}$	$\mathbf{A} \in \mathbb{R}^{N\times N}$ is symmetric and positive semidefinite
$\mathbf{A} \succ \mathbf{0}$	$\mathbf{A} \in \mathbb{R}^{N\times N}$ is symmetric and positive definite
$\\|\mathbf{A}\\|_1$	ℓ_1-norm of $\mathbf{A} \in \mathbb{R}^{N\times M}$, i.e. $\sum_{nm} \|a_{nm}\|$
$\\|\mathbf{A}\\|_\infty$	ℓ_∞-norm of $\mathbf{A} \in \mathbb{R}^{N\times M}$, i.e. $\max_{nm} \|a_{nm}\|$
$\\|\mathbf{A}\\|_2$	spectral norm of $\mathbf{A} \in \mathbb{R}^{N\times N}$, i.e. the maximum eigenvalue of $\mathbf{A} \succ \mathbf{0}$
$\\|\mathbf{A}\\|_{\mathfrak{F}}$	Frobenius norm of $\mathbf{A} \in \mathbb{R}^{N\times M}$, i.e. $\sqrt{\sum_{nm} a_{nm}^2}$
$\mathrm{tr}(\mathbf{A})$	trace of $\mathbf{A} \in \mathbb{R}^{N\times N}$, i.e. $\mathrm{tr}(\mathbf{A}) = \sum_n a_{nn}$
$\langle \mathbf{A}, \mathbf{B} \rangle$	scalar product of $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{N\times M}$, i.e. $\sum_{nm} a_{nm} b_{nm}$
$\mathbf{A}\#\mathbf{B}$	geometric mean of $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{N\times N}$, i.e. the matrix $\mathbf{Z} \succeq \mathbf{0}$ with maximum singular values such that $\begin{bmatrix} \mathbf{A} & \mathbf{Z} \\ \mathbf{Z} & \mathbf{B} \end{bmatrix} \succeq \mathbf{0}$

Gaussian graphical model is simply reflected in the zero entries of the precision matrix $\mathbf{\Omega} = \mathbf{\Sigma}^{-1}$ (Lauritzen, 1996). Let $\mathbf{\Omega} = \{\omega_{n_1 n_2}\}$, two variables n_1 and n_2 are conditionally independent if and only if $\omega_{n_1 n_2} = 0$.

The log-likelihood of a sample $\mathbf{x} \in \mathbb{R}^N$ in a Gaussian graphical model with mean $\boldsymbol{\mu}$ and precision matrix $\mathbf{\Omega}$ is given by:

$$\mathcal{L}(\mathbf{\Omega}, \mathbf{x}) \equiv \log\det\mathbf{\Omega} - (\mathbf{x} - \boldsymbol{\mu})^\top \mathbf{\Omega}(\mathbf{x} - \boldsymbol{\mu}) \tag{1}$$

In this paper, we use a short hand notation for the average log-likelihood of T samples $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(T)} \in \mathbb{R}^N$. Given that this average log-likelihood depends only on the sample covariance matrix $\widehat{\mathbf{\Sigma}}$, we define:

$$\begin{aligned}\mathcal{L}(\mathbf{\Omega}, \widehat{\mathbf{\Sigma}}) &\equiv \frac{1}{T}\sum_t \mathcal{L}(\mathbf{\Omega}, \mathbf{x}^{(t)}) \\ &= \log\det\mathbf{\Omega} - \langle \widehat{\mathbf{\Sigma}}, \mathbf{\Omega} \rangle \end{aligned} \tag{2}$$

A very well known technique for the estimation of precision matrices is Tikhonov regularization (Duchi et al., 2008). Given a sample covariance matrix $\widehat{\mathbf{\Sigma}} \succeq \mathbf{0}$, the *Tikhonov-regularized problem* is defined as:

$$\max_{\mathbf{\Omega} \succ \mathbf{0}} \mathcal{L}(\mathbf{\Omega}, \widehat{\mathbf{\Sigma}}) - \rho\, \mathrm{tr}(\mathbf{\Omega}) \tag{3}$$

for regularization parameter $\rho > 0$. The term $\mathcal{L}(\mathbf{\Omega}, \widehat{\mathbf{\Sigma}})$ is the Gaussian log-likelihood as defined in eq.(2). The optimal solution of eq.(3) is given by:

$$\widehat{\mathbf{\Omega}} = (\widehat{\mathbf{\Sigma}} + \rho\mathbf{I})^{-1} \tag{4}$$

The estimation of sparse precision matrices was first introduced in (Dempster, 1972). It is well known that finding the most sparse precision matrix which fits a dataset is a NP-hard problem (Banerjee et al., 2006; Banerjee et al., 2008). Since the ℓ_1-norm is the tighest convex upper bound of the cardinality of a matrix, several ℓ_1-regularization methods have been proposed. Given a dense sample covariance matrix $\widehat{\mathbf{\Sigma}} \succeq \mathbf{0}$, the *$\ell_1$-regularized problem* is defined as:

$$\max_{\mathbf{\Omega} \succ \mathbf{0}} \mathcal{L}(\mathbf{\Omega}, \widehat{\mathbf{\Sigma}}) - \rho\|\mathbf{\Omega}\|_1 \tag{5}$$

for regularization parameter $\rho > 0$. The term $\mathcal{L}(\mathbf{\Omega}, \widehat{\mathbf{\Sigma}})$ is the Gaussian log-likelihood as defined in eq.(2). The term $\|\mathbf{\Omega}\|_1$ encourages sparseness of the precision matrix.

From a Bayesian viewpoint, eq.(5) is equivalent to maximum likelihood estimation with a prior that assumes that each entry in $\mathbf{\Omega}$ is independent and each of them follow a Laplace distribution (ℓ_1-norm).

Several algorithms have been proposed for solving eq.(5): sparse regression for each variable (Meinshausen & Bühlmann, 2006; Zhou et al., 2011), determinant maximization with linear inequality constraints (Yuan & Lin, 2007), a block coordinate descent method with quadratic programming (Banerjee et al., 2006; Banerjee et al., 2008) or sparse regression (Friedman et al., 2007), a Cholesky decomposition approach (Rothman et al., 2008), a projected gradient method (Duchi et al., 2008), a projected quasi-Newton method (Schmidt et al., 2009), Nesterov's smooth optimization technique (Lu, 2009), an alternating linearization method (Scheinberg et al., 2010), a greedy coordinate descent method (Scheinberg & Rish, 2010), a block coordinate gradient descent method (Yun et al., 2011), quadratic approximation (Hsieh et al., 2011), Newton-like methods (Olsen et al., 2012), iterative thresholding (Guillot et al., 2012), greedy forward and backward steps (Johnson et al., 2012) and divide and conquer (Hsieh et al., 2012).

Additionally, a general rule that uses the sample covariance $\widehat{\mathbf{\Sigma}}$ and splits the graph into connected components was proposed in (Mazumder & Hastie, 2012). After applying this rule, one can use any of the above methods for each component separately. This technique enables the use of ℓ_1-regularization methods to very high dimensional datasets. To the best of our knowledge, (Mazumder & Hastie, 2012) reported results on the highest dimensional dataset (24,481 genes).

3 Problem Setup

In this paper, we assume that the number of variables N is significantly larger than the number of samples T, i.e. $N \gg T$. This is a reasonable assumption is several contexts, for instance in gene expression datasets where N is in the order of 20 thousands to 2 millions, while T is in the order of few tens to few hundreds.

Given a sample covariance matrix $\widehat{\mathbf{\Sigma}} \succeq \mathbf{0}$, we define the *Riccati-regularized problem* as:

$$\max_{\mathbf{\Omega} \succ \mathbf{0}} \mathcal{L}(\mathbf{\Omega}, \widehat{\mathbf{\Sigma}}) - \frac{\rho}{2}\|\mathbf{\Omega}\|_{\mathfrak{F}}^2 \tag{6}$$

for regularization parameter $\rho > 0$. The term $\mathcal{L}(\mathbf{\Omega}, \widehat{\mathbf{\Sigma}})$ is the Gaussian log-likelihood as defined in eq.(2). The term $\|\mathbf{\Omega}\|_{\mathfrak{F}}$ is the Frobenius norm. We chose the name "Riccati regularizer" because the optimal solution of eq.(6) leads to a *Riccati matrix equation.*

From a Bayesian viewpoint, eq.(6) is equivalent to maximum likelihood estimation with a prior that assumes that each entry in $\mathbf{\Omega}$ is independent and each of them follow a Gaussian distribution (ℓ_2^2-norm).

Surprisingly, the problem in eq.(6) has not received much attention. The problem was briefly mentioned in (Witten & Tibshirani, 2009) as a subproblem of a *regression* technique. A tangentially related prior is the ℓ_2^2-norm regularization of the Cholesky factors of the precision matrix (Huang et al., 2006).

It is well known that ℓ_1-regularization allows obtaining an inverse covariance matrix that is "simple" in the sense that it is sparse, with a small number of non-zero entries. Our proposed ℓ_2^2-regularizer allows obtaining an inverse covariance that is also "simple" in the sense that it is low-rank as we will show in Section 4. On the other hand, the solution of the ℓ_1-regularized problem in eq.(5) is full-rank with N different eigenvalues, even for simple cases such as datasets consisting of just two samples. One seemingly unavoidable bottleneck for ℓ_1-regularization is the computation of the covariance matrix $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{N \times N}$ which is $\mathcal{O}(N^2T)$-time and $O(N^2)$-space. Moreover, the work of (Banerjee et al., 2006) showed that in order to obtain an ε-accurate solution, we need $\mathcal{O}(N^{4.5}/\varepsilon)$-time. It is also easy to argue that without further assumptions, sparseness promoting algorithms are $\Omega(N^3)$-time and $O(N^2)$-space. This is prohibitive for very high dimensional datasets. The method of connected components (Mazumder & Hastie, 2012) can reduce the requirement of $\Omega(N^3)$-time. Unfortunately, in order to make the biggest connected component of a reasonable size $N' = 500$ (as prescribed in (Mazumder & Hastie, 2012)), a very large regularization parameter ρ is needed. As a result, the learnt models do not generalize well as we will show experimentally in Section 5.

4 Linear-Time and Space Algorithms

In this section, we propose algorithms with time-complexity $\mathcal{O}(NT^2)$ and space-complexity $\mathcal{O}(NT)$ by using a low-rank parameterization. Given our assumption that $N \gg T$, we can say that our algorithms are linear time and space, i.e. $\mathcal{O}(N)$.

4.1 Solution of the Riccati Problem

Here we show that the solution of our proposed Riccati-regularized problem leads to the Riccati matrix equation, which has a closed form solution.

Theorem 1. *For $\rho > 0$, the optimal solution of the* Riccati-regularized problem *in eq.*(6) *is given by:*

$$\widehat{\mathbf{\Omega}} = \lim_{\varepsilon \to 0^+} \left(\left(\frac{1}{\rho}\widehat{\mathbf{\Sigma}}_\varepsilon\right) \# \left(\widehat{\mathbf{\Sigma}}_\varepsilon^{-1} + \frac{1}{4\rho}\widehat{\mathbf{\Sigma}}_\varepsilon\right) - \frac{1}{2\rho}\widehat{\mathbf{\Sigma}}_\varepsilon \right) \tag{7}$$

where $\widehat{\mathbf{\Sigma}}_\varepsilon = \widehat{\mathbf{\Sigma}} + \varepsilon\mathbf{I}$.

Proof. First, we consider $\widehat{\mathbf{\Sigma}}_\varepsilon = \widehat{\mathbf{\Sigma}} + \varepsilon\mathbf{I} \succ \mathbf{0}$ for some small $\varepsilon > 0$ instead of $\widehat{\mathbf{\Sigma}} \succeq \mathbf{0}$. The ε-corrected version of eq.(6), that is $\max_{\mathbf{\Omega} \succ \mathbf{0}} \mathcal{L}(\mathbf{\Omega}, \widehat{\mathbf{\Sigma}}_\varepsilon) - \frac{\rho}{2}\|\mathbf{\Omega}\|_{\mathfrak{F}}^2$ has the minimizer $\widehat{\mathbf{\Omega}}_\varepsilon$ if and only if the derivative of the objective function at $\mathbf{\Omega} = \widehat{\mathbf{\Omega}}_\varepsilon$ is equal to zero. That is, $\widehat{\mathbf{\Omega}}_\varepsilon^{-1} - \widehat{\mathbf{\Sigma}}_\varepsilon - \rho\widehat{\mathbf{\Omega}}_\varepsilon = \mathbf{0}$. The latter equation is known as the *Riccati matrix equation*, which by virtue of Theorem 3.1 in (Lim, 2006) has the solution $\widehat{\mathbf{\Omega}}_\varepsilon = \left(\frac{1}{\rho}\widehat{\mathbf{\Sigma}}_\varepsilon\right) \# \left(\widehat{\mathbf{\Sigma}}_\varepsilon^{-1} + \frac{1}{4\rho}\widehat{\mathbf{\Sigma}}_\varepsilon\right) - \frac{1}{2\rho}\widehat{\mathbf{\Sigma}}_\varepsilon$. By taking the limit $\varepsilon \to 0^+$, we prove our claim. □

Although the optimal solution in eq.(7) seems to require $\Omega(N^2)$ and $\mathcal{O}(N^3)$ operations, in the next section we show efficient algorithms when $N \gg T$.

4.2 Spectral Method for the Riccati and Tikhonov Problems

In what follows, we provide a method for computing the optimal solution of the Riccati-regularized problem as well as the Tikhonov-regularized problem by using singular value decomposition of the training data. First, we focus on the Riccati-regularized problem.

Theorem 2. *Let $\widehat{\mathbf{\Sigma}} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$ be the singular value decomposition of $\widehat{\mathbf{\Sigma}}$, where $\mathbf{U} \in \mathbb{R}^{N \times T}$ is an orthonormal matrix (i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$) and $\mathbf{D} \in \mathbb{R}^{T \times T}$ is a diagonal matrix. For $\rho > 0$, the optimal solution of the* Riccati-regularized problem *in eq.*(6) *is given by:*

$$\widehat{\mathbf{\Omega}} = \mathbf{U}\widetilde{\mathbf{D}}\mathbf{U}^\top + \frac{1}{\sqrt{\rho}}\mathbf{I} \tag{8}$$

where $\widetilde{\mathbf{D}} \in \mathbb{R}^{T \times T}$ is a diagonal matrix with entries $(\forall t)$ $\widetilde{d}_{tt} = \sqrt{\frac{1}{\rho} + \frac{d_{tt}^2}{4\rho^2}} - \frac{d_{tt}}{2\rho} - \frac{1}{\sqrt{\rho}}$.

Proof Sketch. We consider an *over-complete* singular value decomposition by adding columns to $\mathbf{U}$, thus obtaining $\bar{\mathbf{U}} \in \mathbb{R}^{N \times N}$ such that $\bar{\mathbf{U}}\bar{\mathbf{U}}^\top = \mathbf{I}$, which allows obtaining a singular value decomposition for $\widehat{\mathbf{\Sigma}}_\varepsilon$. Then we apply Theorem 1 and properties of the geometric mean for general as well as diagonal matrices. □

(Please, see Appendix B for detailed proofs.)

Next, we focus on the Tikhonov-regularized problem.

Theorem 3. *Let $\widehat{\mathbf{\Sigma}} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$ be the singular value decomposition of $\widehat{\mathbf{\Sigma}}$, where $\mathbf{U} \in \mathbb{R}^{N\times T}$ is an orthonormal matrix (i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$) and $\mathbf{D} \in \mathbb{R}^{T\times T}$ is a diagonal matrix. For $\rho > 0$, the optimal solution of the* Tikhonov-regularized problem *in eq.(3) is given by:*

$$\widehat{\mathbf{\Omega}} = \mathbf{U}\widetilde{\mathbf{D}}\mathbf{U}^\top + \frac{1}{\rho}\mathbf{I} \tag{9}$$

where $\widetilde{\mathbf{D}} \in \mathbb{R}^{T\times T}$ is a diagonal matrix with entries $(\forall t)$ $\widetilde{d}_{tt} = \frac{-d_{tt}}{\rho(d_{tt}+\rho)}$.

Proof Sketch. We consider an *over-complete* singular value decomposition by adding columns to $\mathbf{U}$, thus obtaining $\bar{\mathbf{U}} \in \mathbb{R}^{N\times N}$ such that $\bar{\mathbf{U}}\bar{\mathbf{U}}^\top = \mathbf{I}$, which allows obtaining a singular value decomposition for $\widehat{\mathbf{\Omega}}$. The final result follows from eq.(4). □

Note that computing $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{N\times N}$ is $\mathcal{O}(N^2T)$-time. In fact, in order to solve the Riccati-regularized problem as well as the Tikhonov-regularized problem, we do not need to compute $\widehat{\mathbf{\Sigma}}$ but its singular value decomposition. The following remark shows that we can obtain the singular value decomposition of $\widehat{\mathbf{\Sigma}}$ by using the singular value decomposition of the training data, which is $\mathcal{O}(NT^2)$-time.

Remark 4. *Let $\mathbf{X} \in \mathbb{R}^{N\times T}$ be a dataset composed of T samples, i.e. $\mathbf{X} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)})$. Without loss of generality, let assume $\mathbf{X}$ has zero mean, i.e. $\widehat{\boldsymbol{\mu}} = \frac{1}{T}\sum_t \mathbf{x}^{(t)} = \mathbf{0}$. The sample covariance matrix is given by $\widehat{\mathbf{\Sigma}} = \frac{1}{T}\mathbf{X}\mathbf{X}^\top \succeq \mathbf{0}$. We can obtain the singular value decomposition of $\widehat{\mathbf{\Sigma}}$ by using the singular value decomposition of $\mathbf{X}$. That is, $\mathbf{X} = \mathbf{U}\widetilde{\mathbf{D}}\mathbf{V}^\top$ where $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{N\times T}$ are orthonormal matrices (i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{V}^\top\mathbf{V} = \mathbf{I}$) and $\widetilde{\mathbf{D}} \in \mathbb{R}^{T\times T}$ is a diagonal matrix. We can recover $\mathbf{D}$ from $\widetilde{\mathbf{D}}$ by using $\mathbf{D} = \frac{1}{T}\widetilde{\mathbf{D}}^2$. Finally, we have $\widehat{\mathbf{\Sigma}} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$.*

The following remark shows that instead of obtaining a single solution for a specific value of the regularization parameter ρ, we can obtain the whole solution path.

Remark 5. *Note that the optimal solutions of the Riccati-regularized problem as well as the Tikhonov-regularized problem have the form $\mathbf{U}\widetilde{\mathbf{D}}\mathbf{U}^\top + c\mathbf{I}$. In these solutions, the diagonal matrix $\widetilde{\mathbf{D}}$ is a function of $\mathbf{D}$ and the regularization parameter ρ, while c is a function of ρ. Therefore, we need to apply the singular value decomposition only once in order to produce a solution for any value of ρ. Furthermore, producing each of those solutions is $\mathcal{O}(T)$-time given that $\widetilde{\mathbf{D}} \in \mathbb{R}^{T\times T}$ is a diagonal matrix.*

4.3 Bounds for the Eigenvalues

Here we show that the optimal solution of the Riccati-regularized problem is well defined (i.e. positive definite with finite eigenvalues).

Corollary 6. *For $\rho > 0$, the optimal solution of the* Riccati-regularized problem *in eq.(6) is bounded as follows:*

$$\alpha\mathbf{I} \preceq \widehat{\mathbf{\Omega}} \preceq \beta\mathbf{I} \tag{10}$$

where $\alpha = \sqrt{\frac{1}{\rho} + \frac{1}{4\rho^2}\|\widehat{\mathbf{\Sigma}}\|_2^2} - \frac{1}{2\rho}\|\widehat{\mathbf{\Sigma}}\|_2$ and $\beta = \frac{1}{\sqrt{\rho}}$. That is, the eigenvalues of the optimal solution $\widehat{\mathbf{\Omega}}$ are in the range $[\alpha; \beta]$ and therefore, $\widehat{\mathbf{\Omega}}$ is a well defined precision matrix.

Proof Sketch. Theorem 2 gives the eigendecomposition of the solution $\widehat{\mathbf{\Omega}}$. That is, the diagonal $\widetilde{\mathbf{D}} + \frac{1}{\sqrt{\rho}}\mathbf{I}$ contains the eigenvalues of $\widehat{\mathbf{\Omega}}$. Then we bound the values in the diagonal. □

Remark 7. *We can obtain bounds for the* Tikhonov-regularized problem *in eq.(3) by arguments similar to the ones in Corollary 6. That is, the eigenvalues of the optimal solution $\widehat{\mathbf{\Omega}}$ are in the range $[\alpha; \beta]$ where $\alpha = \frac{1}{\|\widehat{\mathbf{\Sigma}}\|_2 + \rho}$ and $\beta = \frac{1}{\rho}$.*

4.4 Logarithmic Sample Complexity

The ℓ_1 regularization for Gaussian graphical models has a sample complexity that scales logarithmically with respect to the number of variables N (Ravikumar et al., 2011). On the other hand, it is known that in general, ℓ_2^2 regularization has a worst-case sample complexity that scales linearly with respect to the number of variables, for problems such as classification (Ng, 2004) and compressed sensing (Cohen et al., 2009).

In this section, we prove the logarithmic sample complexity of the Riccati problem under the *spiked covariance model* originally proposed by (Johnstone, 2001). Without loss of generality, we assume zero mean as in (Ravikumar et al., 2011). First we present the generative model.

Definition 8. *The* spiked covariance model *for N variables and $K \ll N$ components is a generative model $\mathcal{P}$ parameterized by $(\mathbf{U}, \mathbf{D}, \beta)$, where $\mathbf{U} \in \mathbb{R}^{N\times K}$ is an orthonormal matrix (i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$), $\mathbf{D} \in \mathbb{R}^{K\times K}$ is a positive diagonal matrix, and $\beta > 0$. A sample $\mathbf{x} \in \mathbb{R}^N$ is generated by first sampling the independent random vectors $\mathbf{y} \in \mathbb{R}^K$ and $\boldsymbol{\xi} \in \mathbb{R}^N$, both* *with uncorrelated sub-Gaussian entries with zero mean and unit variance. Then we generate the sample* $\mathbf{x}$ *as follows:*

$$\mathbf{x} = \mathbf{U}\mathbf{D}^{1/2}\mathbf{y} + \sqrt{\frac{\beta}{N}}\boldsymbol{\xi} \tag{11}$$

Furthermore, it is easy to verify that:

$$\boldsymbol{\Sigma} = \mathbb{E}_{\mathcal{P}}[\mathbf{x}\mathbf{x}^\top] = \mathbf{U}\mathbf{D}\mathbf{U}^\top + \frac{\beta}{N}\mathbf{I} \tag{12}$$

(Please, see Appendix B for a detailed proof.)

Note that the above model is not a Gaussian distribution in general. It is Gaussian if and only if the random variables $\mathbf{y}$ and $\boldsymbol{\xi}$ are Gaussians.

Next we present a concentration inequality for the approximation of the ground truth covariance.

Lemma 9. *Let $\mathcal{P}$ be a ground truth* spiked covariance model *for N variables and $K \ll N$ components, parameterized by $(\mathbf{U}^*, \mathbf{D}^*, \beta^*)$ with covariance $\boldsymbol{\Sigma}^*$ as in Definition 8. Given a sample covariance matrix $\widehat{\boldsymbol{\Sigma}}$ computed from T samples $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)} \in \mathbb{R}^N$ drawn independently from $\mathcal{P}$. With probability at least $1-\delta$, the sample covariance fulfills the following concentration inequality:*

$$\|\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}^*\|_{\mathfrak{F}} \leq 40(K\sqrt{\|\mathbf{D}^*\|_{\mathfrak{F}}} + \sqrt{\beta^*})^2 \times \sqrt{\frac{4\log(N+K) + 2\log\frac{4}{\delta}}{T}} \tag{13}$$

Proof Sketch. By the definition of the sample covariance, matrix norm inequalities and a concentration inequality for the ℓ_∞-norm of a sample covariance matrix from sub-Gaussian random variables given by Lemma 1 in (Ravikumar et al., 2011). □

Armed with the previous result, we present our logarithmic sample complexity for the Riccati problem under the *spiked covariance model.*

Theorem 10. *Let $\mathcal{P}$ be a ground truth* spiked covariance model *for N variables and $K \ll N$ components, parameterized by $(\mathbf{U}^*, \mathbf{D}^*, \beta^*)$ with covariance $\boldsymbol{\Sigma}^*$ as in Definition 8. Let $\mathcal{Q}^*$ be the distribution generated by a Gaussian graphical model with zero mean and precision matrix $\boldsymbol{\Omega}^* = \boldsymbol{\Sigma}^{*-1}$. That is, $\mathcal{Q}^*$ is the projection of $\mathcal{P}$ to the family of Gaussian distributions. Given a sample covariance matrix $\widehat{\boldsymbol{\Sigma}}$ computed from T samples $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)} \in \mathbb{R}^N$ drawn independently from $\mathcal{P}$. Let $\widehat{\mathcal{Q}}$ be the distribution generated by a Gaussian graphical model with zero mean and precision matrix $\widehat{\boldsymbol{\Omega}}$, the solution of the* Riccati-regularized problem *in eq.(6). With probability at least $1-\delta$, we have:*

$$\mathcal{KL}(\mathcal{P}||\widehat{\mathcal{Q}}) - \mathcal{KL}(\mathcal{P}||\mathcal{Q}^*) = \mathbb{E}_{\mathcal{P}}[\mathcal{L}(\boldsymbol{\Omega}^*, \mathbf{x}) - \mathcal{L}(\widehat{\boldsymbol{\Omega}}, \mathbf{x})]$$
$$\leq 40(K\sqrt{\|\mathbf{D}^*\|_{\mathfrak{F}}} + \sqrt{\beta^*})^2 \sqrt{\frac{4\log(N+K) + 2\log\frac{4}{\delta}}{T}} \times \left(\tfrac{1}{4} + \|\boldsymbol{\Omega}^*\|_{\mathfrak{F}} + \|\boldsymbol{\Omega}^*\|_{\mathfrak{F}}^2\right) \tag{14}$$

where $\mathcal{L}(\boldsymbol{\Omega}, \mathbf{x})$ is the Gaussian log-likelihood as defined in eq.(1).

Proof. Let $p(\mathbf{x})$, $\widehat{q}(\mathbf{x})$ and $q^*(\mathbf{x})$ be the probability density functions of $\mathcal{P}$, $\widehat{\mathcal{Q}}$ and $\mathcal{Q}^*$ respectively. Note that $\mathcal{KL}(\mathcal{P}||\widehat{\mathcal{Q}}) = \mathbb{E}_{\mathcal{P}}[\log p(\mathbf{x}) - \log \widehat{q}(\mathbf{x})] = -\mathcal{H}(\mathcal{P}) - \mathbb{E}_{\mathcal{P}}[\mathcal{L}(\widehat{\boldsymbol{\Omega}}, \mathbf{x})]$. Similarly, $\mathcal{KL}(\mathcal{P}||\mathcal{Q}^*) = -\mathcal{H}(\mathcal{P}) - \mathbb{E}_{\mathcal{P}}[\mathcal{L}(\boldsymbol{\Omega}^*, \mathbf{x})]$. Therefore, $\mathcal{KL}(\mathcal{P}||\widehat{\mathcal{Q}}) - \mathcal{KL}(\mathcal{P}||\mathcal{Q}^*) = \mathbb{E}_{\mathcal{P}}[\mathcal{L}(\boldsymbol{\Omega}^*, \mathbf{x}) - \mathcal{L}(\widehat{\boldsymbol{\Omega}}, \mathbf{x})]$.

By using the definition of the Gaussian log-likelihood in eq.(1), the expected log-likelihood is given by:

$$\mathcal{L}^*(\boldsymbol{\Omega}) \equiv \mathbb{E}_{\mathcal{P}}[\mathcal{L}(\boldsymbol{\Omega}, \mathbf{x})] = \log\det\boldsymbol{\Omega} - \langle \boldsymbol{\Sigma}^*, \boldsymbol{\Omega} \rangle$$

Similarly, define the shorthand notation for the finite-sample log-likelihood in eq.(2) as follows:

$$\widehat{\mathcal{L}}(\boldsymbol{\Omega}) \equiv \mathcal{L}(\boldsymbol{\Omega}, \widehat{\boldsymbol{\Sigma}}) = \log\det\boldsymbol{\Omega} - \langle \widehat{\boldsymbol{\Sigma}}, \boldsymbol{\Omega} \rangle$$

From the above definitions we have:

$$\mathcal{L}^*(\boldsymbol{\Omega}) - \widehat{\mathcal{L}}(\boldsymbol{\Omega}) = \langle \widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}^*, \boldsymbol{\Omega} \rangle$$

By the Cauchy-Schwarz inequality and Lemma 9:

$$|\mathcal{L}^*(\boldsymbol{\Omega}) - \widehat{\mathcal{L}}(\boldsymbol{\Omega})| \leq \|\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}^*\|_{\mathfrak{F}} \|\boldsymbol{\Omega}\|_{\mathfrak{F}}$$
$$\leq \gamma \|\boldsymbol{\Omega}\|_{\mathfrak{F}}$$

for $\gamma = 40(K\sqrt{\|\mathbf{D}^*\|_{\mathfrak{F}}} + \sqrt{\beta^*})^2 \sqrt{\frac{4\log(N+K)+2\log\frac{4}{\delta}}{T}}$. Thus, we obtained a “uniform convergence” statement for the loss.

Note that $\widehat{\boldsymbol{\Omega}}$ (the minimizer of the regularized finite-sample loss) and $\boldsymbol{\Omega}^*$ (the minimizer of the expected loss) fulfill by definition:

$$\widehat{\boldsymbol{\Omega}} = \arg\max_{\boldsymbol{\Omega}} \widehat{\mathcal{L}}(\boldsymbol{\Omega}) - \frac{\rho}{2}\|\boldsymbol{\Omega}\|_{\mathfrak{F}}^2$$
$$\boldsymbol{\Omega}^* = \arg\max_{\boldsymbol{\Omega}} \mathcal{L}^*(\boldsymbol{\Omega})$$

Therefore, by optimality of $\widehat{\boldsymbol{\Omega}}$ we have:

$$\widehat{\mathcal{L}}(\boldsymbol{\Omega}^*) - \frac{\rho}{2}\|\boldsymbol{\Omega}^*\|_{\mathfrak{F}}^2 \leq \widehat{\mathcal{L}}(\widehat{\boldsymbol{\Omega}}) - \frac{\rho}{2}\|\widehat{\boldsymbol{\Omega}}\|_{\mathfrak{F}}^2$$
$$\Rightarrow \widehat{\mathcal{L}}(\boldsymbol{\Omega}^*) - \widehat{\mathcal{L}}(\widehat{\boldsymbol{\Omega}}) \leq \frac{\rho}{2}\|\boldsymbol{\Omega}^*\|_{\mathfrak{F}}^2 - \frac{\rho}{2}\|\widehat{\boldsymbol{\Omega}}\|_{\mathfrak{F}}^2$$

By using the “uniform convergence” statement, the above results and setting $\rho = 2\gamma$:

$$\mathcal{L}^*(\boldsymbol{\Omega}^*) - \mathcal{L}^*(\widehat{\boldsymbol{\Omega}}) \leq \widehat{\mathcal{L}}(\boldsymbol{\Omega}^*) - \widehat{\mathcal{L}}(\widehat{\boldsymbol{\Omega}}) + \gamma\|\boldsymbol{\Omega}^*\|_{\mathfrak{F}} + \gamma\|\widehat{\boldsymbol{\Omega}}\|_{\mathfrak{F}}$$
$$\leq \frac{\rho}{2}\|\boldsymbol{\Omega}^*\|_{\mathfrak{F}}^2 - \frac{\rho}{2}\|\widehat{\boldsymbol{\Omega}}\|_{\mathfrak{F}}^2 + \gamma\|\boldsymbol{\Omega}^*\|_{\mathfrak{F}} + \gamma\|\widehat{\boldsymbol{\Omega}}\|_{\mathfrak{F}}$$
$$= \gamma(\|\widehat{\boldsymbol{\Omega}}\|_{\mathfrak{F}} - \|\widehat{\boldsymbol{\Omega}}\|_{\mathfrak{F}}^2 + \|\boldsymbol{\Omega}^*\|_{\mathfrak{F}} + \|\boldsymbol{\Omega}^*\|_{\mathfrak{F}}^2)$$

By noting that $\|\widehat{\mathbf{\Omega}}\|_{\mathfrak{F}} - \|\widehat{\mathbf{\Omega}}\|_{\mathfrak{F}}^2 \leq \frac{1}{4}$, we prove our claim. $\square$

4.5 Spectral Method for the Sparse Problem

In this section, we propose sparsification of the dense solution and show an upper bound of the degradation of the Kullback-Leibler divergence and expected log-likelihood.

First, we study the relationship between the sparseness of the low-rank parameterization and the generated precision matrix. We analyze the expected value of densities of random matrices. We assume that each entry in the matrix is statistically independent, which is not a novel assumption. From a Bayesian viewpoint, the ℓ_1-regularized problem in eq.(5) assumes that each entry in the precision matrix $\mathbf{\Omega}$ is independent. We also made a similar assumption for our Riccati-regularized problem in eq.(6).

Lemma 11. *Let $\mathbf{A} \in \mathbb{R}^{N \times T}$ be a random matrix with statistically independent entries and expected density p, i.e. $(\forall n, t)\ \mathbb{P}[a_{nt} \neq 0] = p$. Furthermore, assume that conditional distribution of $a_{nt} \mid a_{nt} \neq 0$ has a domain with non-zero Lebesgue measure. Let $\mathbf{D} \in \mathbb{R}^{T \times T}$ be a diagonal matrix such that $(\forall t)\ d_{tt} \neq 0$. Let c be an arbitrary real value. Let $\mathbf{B} = \mathbf{A}\mathbf{D}\mathbf{A}^\top + c\mathbf{I}$. The expected non-diagonal density of $\mathbf{B} \in \mathbb{R}^{N \times N}$ is given by:*

$$(\forall n_1 \neq n_2)\ \mathbb{P}[b_{n_1 n_2} \neq 0] = 1 - (1 - p^2)^T \qquad (15)$$

Proof Sketch. Note that for $n_1 \neq n_2$, we have $b_{n_1 n_2} = \sum_t d_{tt} a_{n_1 t} a_{n_2 t}$. We argue that the probability that $b_{n_1 n_2} \neq 0$ is equal to the probability that $a_{n_1 t} \neq 0$ and $a_{n_2 t} \neq 0$ for at least one t. The final result follows from independence of the entries of $\mathbf{A}$. $\square$

It is easy to construct (non-random) specific instances in which the density of $\mathbf{A}$ and $\mathbf{B}$ are unrelated. Consider two examples in which $\mathbf{D} = \mathbf{I}$, $c = 0$ and therefore $\mathbf{B} = \mathbf{A}\mathbf{A}^\top$. As a first example, let $\mathbf{A}$ have zero entries except for $(\forall n)\ a_{n1} = 1$. That is, $\mathbf{A}$ is very sparse but $\mathbf{B} = \mathbf{A}\mathbf{A}^\top = \mathbf{1}\mathbf{1}^\top$ is dense. As a second example, let $\mathbf{B}$ have zero entries except for $(\forall n)\ b_{n1} = b_{1n} = 1, b_{nn} = N$. That is, $\mathbf{B} = \mathbf{A}\mathbf{A}^\top$ is very sparse but its Chokesly decomposition $\mathbf{A}$ is dense.

Next, we show that sparsification of a dense precision matrix produces a precision matrix that is close to the original one. Furthermore, we show that such matrix is positive definite.

Theorem 12. *Let $\mathbf{\Omega} = \mathbf{U}\mathbf{D}\mathbf{U}^\top + \beta\mathbf{I}$ be a precision matrix, where $\mathbf{U} \in \mathbb{R}^{N \times T}$ is an orthonormal matrix (i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$), $\beta > \alpha > 0$ and $\mathbf{D} \in \mathbb{R}^{T \times T}$ is a negative diagonal matrix such that $(\forall t)\ -(\beta - \alpha) < d_{tt} \leq 0$ (or equivalently $\alpha\mathbf{I} \preceq \mathbf{\Omega} \preceq \beta\mathbf{I}$). Let $\widetilde{\mathbf{U}} \in \mathbb{R}^{N \times T}$ be a sparse matrix constructed by soft-thresholding:*

$$(\forall n, t)\ \widetilde{u}_{nt} = \operatorname{sign}(u_{nt}) \max\left(0, |u_{nt}| - \tfrac{\lambda}{\sqrt{NT}}\right) \qquad (16)$$

or by hard-thresholding:

$$(\forall n, t)\ \widetilde{u}_{nt} = u_{nt} 1[|u_{nt}| \geq \tfrac{\lambda}{\sqrt{NT}}] \qquad (17)$$

for some $\lambda > 0$. The sparse *precision matrix $\widetilde{\mathbf{\Omega}} = \widetilde{\mathbf{U}}\mathbf{D}\widetilde{\mathbf{U}}^\top + \beta\mathbf{I}$ satisfies:*

$$\|\widetilde{\mathbf{\Omega}} - \mathbf{\Omega}\|_2 \leq (2\lambda + \lambda^2)(\beta - \alpha) \qquad (18)$$

Furthermore, $\widetilde{\mathbf{\Omega}}$ preserves positive definiteness of the dense *precision matrix $\mathbf{\Omega}$. More formally:*

$$\alpha\mathbf{I} \preceq \widetilde{\mathbf{\Omega}} \preceq \beta\mathbf{I} \qquad (19)$$

That is, the eigenvalues of $\widetilde{\mathbf{\Omega}}$ are in the range $[\alpha; \beta]$ and therefore, $\widetilde{\mathbf{\Omega}}$ is a well defined sparse *precision matrix.*

Proof Sketch. The claims in eq.(18) and eq.(19) follow from matrix norm inequalities. Additionally, eq.(19) follows from showing that $\widetilde{\mathbf{U}}\mathbf{D}\widetilde{\mathbf{U}}^\top$ is negative semidefinite and that $\|\widetilde{\mathbf{U}}\|_2 \leq \|\mathbf{U}\|_2 = 1$. $\square$

Finally, we derive an upper bound of the degradation of the Kullback-Leibler divergence and expected log-likelihood. Thus, sparsification of a dense solution has provable performance guarantees.

Theorem 13. *Let $\mathcal{P}$ be the ground truth distribution of the data sample $\mathbf{x}$. Let $\mathcal{Q}$ be the distribution generated by a Gaussian graphical model with mean $\boldsymbol{\mu}$ and precision matrix $\mathbf{\Omega}$. Let $\widetilde{\mathcal{Q}}$ be the distribution generated by a Gaussian graphical model with mean $\boldsymbol{\mu}$ and precision matrix $\widetilde{\mathbf{\Omega}}$. Assume $\alpha\mathbf{I} \preceq \mathbf{\Omega}$ and $\alpha\mathbf{I} \preceq \widetilde{\mathbf{\Omega}}$ for $\alpha > 0$. The Kullback-Leibler divergence from $\mathcal{P}$ to $\widetilde{\mathcal{Q}}$ is close to the Kullback-Leibler divergence from $\mathcal{P}$ to $\mathcal{Q}$. More formally:*

$$\begin{aligned} &\mathcal{KL}(\mathcal{P}||\widetilde{\mathcal{Q}}) - \mathcal{KL}(\mathcal{P}||\mathcal{Q}) = \mathbb{E}_{\mathcal{P}}[\mathcal{L}(\mathbf{\Omega}, \mathbf{x}) - \mathcal{L}(\widetilde{\mathbf{\Omega}}, \mathbf{x})] \\ &\leq \left(\frac{1}{\alpha} + \|\mathbb{E}_{\mathcal{P}}[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^\top]\|_2\right) \|\widetilde{\mathbf{\Omega}} - \mathbf{\Omega}\|_2 \end{aligned} \qquad (20)$$

where $\mathcal{L}(\mathbf{\Omega}, \mathbf{x})$ is the Gaussian log-likelihood as defined in eq.(1). Moreover, the above bound is finite provided that $\|\mathbb{E}_{\mathcal{P}}[\mathbf{x}]\|_2$ and $\|\mathbb{E}_{\mathcal{P}}[\mathbf{x}\mathbf{x}^\top]\|_2$ are finite.

Proof Sketch. By definition of the Kullback-Leibler divergence, the Gaussian log-likelihood in eq.(1) and by showing that the resulting expression is Lipschitz continuous with respect to the spectral norm. $\square$

4.6 Using the Learnt Models in Linear-Time

Next we provide techniques for using our learnt models, such as removing unimportant variables, computing likelihoods and conditional distributions.

Removal of Unimportant Variables. Recall that the final goal for a data analyst is not only to be able to learn models from data but also to browse such learnt models in order to discover "important" interactions. We define the *unimportant variable set* as the set of variables in which the absolute value of every *partial correlation* is lower than a specified threshold.

Definition 14. *An* unimportant variable set *of a Gaussian graphical model defined by the precision matrix* $\mathbf{\Omega} = \{\omega_{n_1 n_2}\}$ *for threshold* $\varepsilon > 0$ *is a set* $\mathcal{S} \subseteq \{1, \ldots, N\}$ *such that:*

$$(\forall n_1, n_2 \in \mathcal{S})\ \frac{|\omega_{n_1 n_2}|}{\sqrt{\omega_{n_1 n_1}\omega_{n_2 n_2}}} \leq \varepsilon \tag{21}$$

Note that our definition does not require that the size of $\mathcal{S}$ is maximal. Here we provide a technique that discovers the *unimportant variable set* in linear-time.

Lemma 15. *Let* $\mathbf{\Omega} = \mathbf{A}\mathbf{D}\mathbf{A}^\top + c\mathbf{I}$ *be a precision matrix, where* $\mathbf{A} \in \mathbb{R}^{N\times T}$ *is an arbitrary matrix,* $\mathbf{D} \in \mathbb{R}^{T\times T}$ *is an arbitrary diagonal matrix, and* c *is an arbitrary real value, such that* $\mathbf{\Omega}$ *is a well defined precision matrix (i.e.* $\mathbf{\Omega} \succ \mathbf{0}$*). An* unimportant variable set *of the Gaussian graphical model defined by* $\mathbf{\Omega} = \{\omega_{n_1 n_2}\}$ *for threshold* $\varepsilon > 0$ *can be detected by applying the rule:*

$$(\forall n_1, n_2 \in \mathcal{S})\ \frac{|\omega_{n_1 n_2}|}{\sqrt{\omega_{n_1 n_1}\omega_{n_2 n_2}}} \leq \min\left(q(n_1), q(n_2)\right) \leq \varepsilon \tag{22}$$

where $q(n_1) = \frac{\sum_t |d_{tt}a_{n_1 t}| \max_{n_2} |a_{n_2 t}|}{\sqrt{r(n_1)\min_{n_2} r(n_2)}}$ *and* $r(n) = \sum_t d_{tt}a_{nt}^2 + c$*. Furthermore, this operation has time-complexity* $\mathcal{O}(NT)$ *when applied to all variables in* $\{1, \ldots, N\}$*.*

Proof Sketch. Straightforward, from algebra. □

Computation of the Likelihood. In tasks such as classification, we would assign a given sample to the class whose model produced the highest likelihood among all classes. Here we provide a method for computing the likelihood of a given sample in the learnt model.

Lemma 16. *Let* $\mathbf{\Omega} = \mathbf{A}\mathbf{D}\mathbf{A}^\top + c\mathbf{I}$ *be a precision matrix, where* $\mathbf{A} \in \mathbb{R}^{N\times T}$ *is an arbitrary matrix,* $\mathbf{D} \in \mathbb{R}^{T\times T}$ *is an arbitrary diagonal matrix, and* c *is an arbitrary real value, such that* $\mathbf{\Omega}$ *is a well defined precision matrix (i.e.* $\mathbf{\Omega} \succ \mathbf{0}$*). The log-likelihood of a sample* $\mathbf{x}$ *in a Gaussian graphical model with mean* $\boldsymbol{\mu}$ *and precision matrix* $\mathbf{\Omega}$ *is given by:*

$$\begin{aligned}\mathcal{L}(\mathbf{\Omega}, \mathbf{x}) = &\log\det\left(\mathbf{I} + \frac{1}{c}\mathbf{A}^\top\mathbf{A}\mathbf{D}\right) + N\log c \\ &- ((\mathbf{x}-\boldsymbol{\mu})^\top\mathbf{A})\mathbf{D}(\mathbf{A}^\top(\mathbf{x}-\boldsymbol{\mu})) \\ &- c(\mathbf{x}-\boldsymbol{\mu})^\top(\mathbf{x}-\boldsymbol{\mu})\end{aligned} \tag{23}$$

where $\mathcal{L}(\mathbf{\Omega}, \mathbf{x})$ *is the Gaussian log-likelihood as defined in eq.*(1)*. Furthermore, this operation has time-complexity* $\mathcal{O}(NT^2)$*.*

Proof Sketch. By the matrix determinant lemma. □

Conditional Distributions. In some contexts, it is important to perform inference for some unobserved variables when given some observed variables. Here we provide a method for computing the conditional distribution that takes advantage of our low-rank parameterization. First, we show how to transform from a non-orthonormal parameterization to an orthonormal one.

Remark 17. *Let* $\mathbf{A} \in \mathbb{R}^{N\times T}$ *be an arbitrary matrix. Let* $\mathbf{D} \in \mathbb{R}^{T\times T}$ *be a negative diagonal matrix (i.e.* $(\forall t)\ d_{tt} < 0$*). Let* $\mathbf{B} = \mathbf{A}\mathbf{D}\mathbf{A}^\top$*. Since* $\mathbf{B}$ *has rank at most* T*, we can compute its singular value decomposition* $\mathbf{B} = \mathbf{U}\widetilde{\mathbf{D}}\mathbf{U}^\top$ *where* $\mathbf{U} \in \mathbb{R}^{N\times T}$ *is an orthonormal matrix (i.e.* $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$*) and* $\widetilde{\mathbf{D}} \in \mathbb{R}^{T\times T}$ *is a negative diagonal matrix (i.e.* $(\forall t)\ \widetilde{d}_{tt} < 0$*). In order to do this, we compute the singular value decomposition of* $\mathbf{A}(-\mathbf{D})^{1/2} = \mathbf{U}\widetilde{\mathbf{D}}\mathbf{V}^\top$*. We can recover* $\mathbf{D}$ *from* $\widetilde{\mathbf{D}}$ *by using* $\mathbf{D} = -\widetilde{\mathbf{D}}^2$*.*

Given the previous observation, our computation of conditional distributions will focus only on orthonormal parameterizations.

Lemma 18. *Let* $\mathbf{\Omega} = \mathbf{U}\mathbf{D}\mathbf{U}^\top + c\mathbf{I}$ *be a precision matrix, where* $\mathbf{U} \in \mathbb{R}^{N\times T}$ *is an orthonormal matrix (i.e.* $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$*),* $\mathbf{D} \in \mathbb{R}^{T\times T}$ *is an arbitrary diagonal matrix, and* c *is an arbitrary real value, such that* $\mathbf{\Omega}$ *is a well defined precision matrix (i.e.* $\mathbf{\Omega} \succ \mathbf{0}$*). Assume that* $\mathbf{x}$ *follows a Gaussian graphical model with mean* $\boldsymbol{\mu}$ *and precision matrix* $\mathbf{\Omega}$*, and assume that we partition the variables into two sets as follows:*

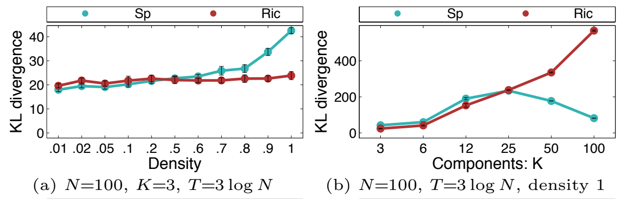
$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix}\boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2\end{bmatrix}, \mathbf{U} = \begin{bmatrix}\mathbf{U}_1 \\ \mathbf{U}_2\end{bmatrix} \tag{24}$$

The conditional distribution of $\mathbf{x}_1 \mid \mathbf{x}_2$ *is a Gaussian graphical model with mean* $\boldsymbol{\mu}_{1|2}$ *and precision matrix* $\mathbf{\Omega}_{11}$*, such that:*

$$\begin{aligned}&\text{i. } \boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 - \mathbf{U}_1\widetilde{\mathbf{D}}\mathbf{U}_2^\top(\mathbf{x}_2 - \boldsymbol{\mu}_2) \\ &\text{ii. } \mathbf{\Omega}_{11} = \mathbf{U}_1\mathbf{D}\mathbf{U}_1^\top + c\mathbf{I}\end{aligned} \tag{25}$$

where $\widetilde{\mathbf{D}} \in \mathbb{R}^{T\times T}$ *is a diagonal matrix with entries* $(\forall t)\ \widetilde{d}_{tt} = \frac{d_{tt}}{d_{tt}+c}$*.*

Proof Sketch. By definition of conditional distributions of Gaussians (Lauritzen, 1996) and Theorem 3. □


(a) N=100, K=3, T=3 log N (b) N=100, T=3 log N, density 1

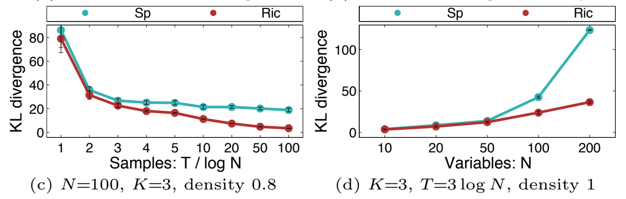

(c) N=100, K=3, density 0.8 (d) K=3, T=3 log N, density 1

Figure 1: Kullback-Leibler divergence between the ground truth *spiked covariance model* and the learnt models for different (a) inverse covariance densities, (b) number of components, (c) samples and (d) variables (error bars shown at 90% confidence interval). The Riccati (Ric) method performs better for ground truth with moderate to high density, while the sparse method (Sp) performs better for low densities. The sparse method performs better for ground truth with large number of components, while the Riccati method performs better for small number of components. The behavior of both methods is similar with respect to the number of samples. The sparse method degrades more than the Riccati method with respect to the number of variables.

Table 2: Gene expression datasets.

Dataset	Disease	Samples	Variables
GSE1898	Liver cancer	182	21,794
GSE29638	Colon cancer	50	22,011
GSE30378	Colon cancer	95	22,011
GSE20194	Breast cancer	278	22,283
GSE22219	Breast cancer	216	24,332
GSE13294	Colon cancer	155	54,675
GSE17951	Prostate cancer	154	54,675
GSE18105	Colon cancer	111	54,675
GSE1476	Colon cancer	150	59,381
GSE14322	Liver cancer	76	104,702
GSE18638	Colon cancer	98	235,826
GSE33011	Ovarian cancer	80	367,657
GSE30217	Leukemia	54	964,431
GSE33848	Lung cancer	30	1,852,426

Table 3: Runtimes for gene expression datasets. Our Riccati method is considerably faster than sparse method.

Dataset	Sparse	Riccati
GSE1898,29638,30378,20194,22219	3.8 min	1.2 sec
GSE13294,17951,18105,1476	14.9 min	1.6 sec
GSE14322	30.4 min	1.0 sec
GSE18638	3.1 hr	2.6 sec
GSE33011	6.0 hr	2.5 sec
GSE30217	1.3 days	3.8 sec
GSE33848	5.4 days	2.8 sec

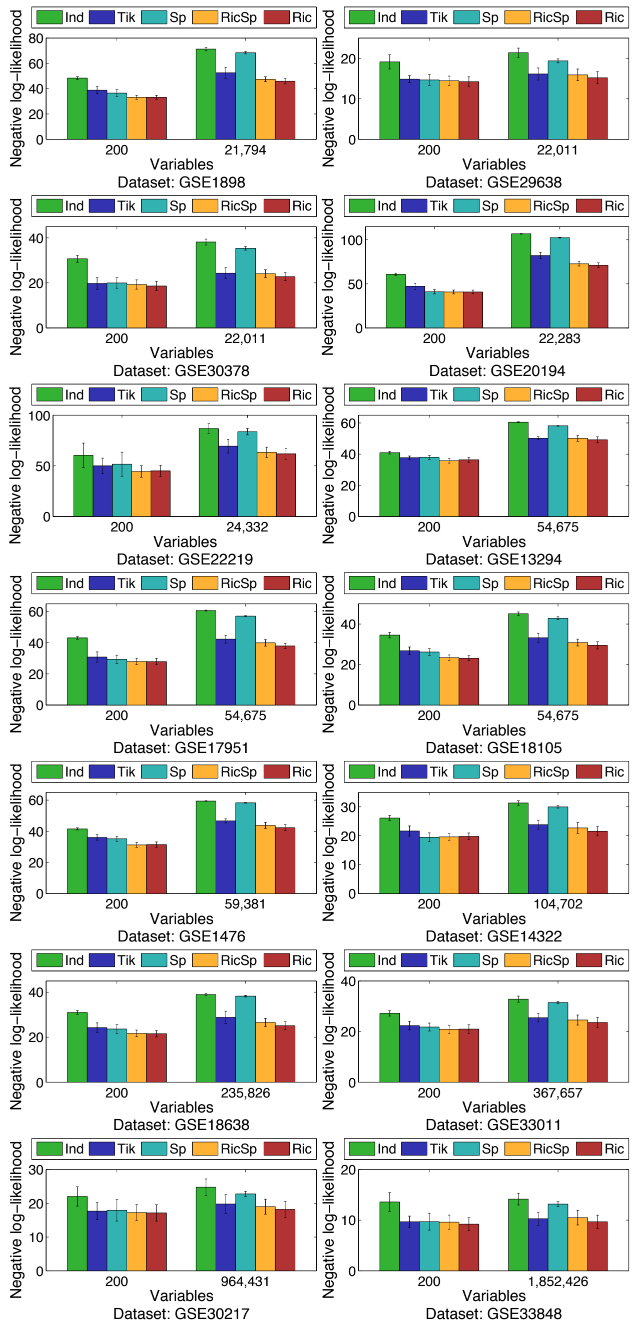


Figure 2: Negative test log-likelihood for gene expression datasets (error bars shown at 90% confidence interval). Our Riccati (Ric) and sparse Riccati (RicSp) methods perform better than the sparse method (Sp), Tikhonov method (Tik) and the fully independent model (Ind). Since the method of (Mazumder & Hastie, 2012) requires high regularization for producing reasonably sized components, the performance of the sparse method (Sp) when using all the variables degrade considerably when compared to 200 variables.

5 Experimental Results

We begin with a synthetic example to test the ability of the method to recover the ground truth distribution from data. We used the *spiked covariance model* as in Definition 8 for $N = 100$ variables, $K = 3$ components and $\beta = 1$. Additionally, we control for the density of the related ground truth inverse covariance matrix. We performed 20 repetitions. For each repetition, we generate a different ground truth model at random.

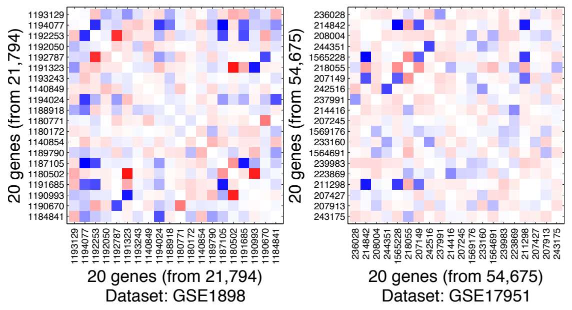


Figure 3: Genes with "important" interactions for two gene expression datasets analyzed with our Riccati method.

We then produce $T = 3 \log N$ samples for training and the same number of samples for validation. In our experiments, we test different scenarios, by varying the density of the ground truth inverse covariance, the number of components, samples and variables.

In Figure 1, we report the Kullback-Leibler divergence between the ground truth and the learnt models. For learning sparse models, we used the method of (Friedman et al., 2007). The Riccati method performs better for ground truth with moderate to high density, while the sparse method performs better for low densities. The sparse method performs better for ground truth with large number of components (as expected when $K \in \mathcal{O}(N)$), while the Riccati method performs better for small number of components (as expected when $K \ll N$). The behavior of both methods is similar with respect to the number of samples. The sparse method degrades more than the Riccati method with respect to the number of variables.

For experimental validation on real-world datasets, we used 14 cancer datasets publicly available at the Gene Expression Omnibus (`http://www.ncbi.nlm.nih.gov/geo/`). Table 2 shows the datasets as well as the number of samples and variables on each of them. We preprocessed the data so that each variable is zero mean and unit variance across the dataset. We performed 50 repetitions. For each repetition, we used one third of the samples for training, one third for validation and the remaining third for testing. We report the negative log-likelihood on the testing set (after subtracting the entropy measured on the testing set in order to make it comparable to the Kullback-Leibler divergence).

Since regular sparseness promoting methods do not scale to our setting of more than 20 thousands variables, we validate our method in two regimes. In the first regime, for each of the 50 repetitions, we select $N = 200$ variables uniformly at random and use the sparse method of (Friedman et al., 2007). In the second regime, we use all the variables in the dataset, and use the method of (Mazumder & Hastie, 2012) so that the biggest connected component has at most $N' = 500$ variables (as prescribed in (Mazumder & Hastie, 2012)). The technique of (Mazumder & Hastie, 2012) computes a graph from edges with an absolute value of the covariance higher than the regularization parameter ρ and then splits the graph into its connected components. Since the whole sample covariance matrix could not fit in memory, we computed it in batches of rows (Mazumder & Hastie, 2012). Unfortunately, in order to make the biggest connected component of a reasonable size $N' = 500$ (as prescribed in (Mazumder & Hastie, 2012)), a high value of ρ is needed. In order to circumvent this problem, we used a high value of ρ for splitting the graph into its connected components, and allowed for low values of ρ for computing the precision matrix for each component. For our sparse Riccati method, we used soft-thresholding of the Riccati solution with $\lambda = \rho$.

In Figure 2, we observe that our Riccati and sparse Riccati methods perform better than the comparison methods. Since the method of (Mazumder & Hastie, 2012) requires high regularization for producing reasonably sized components, the performance of the sparse method when using all the variables degrade considerably when compared to 200 variables.

In Figure 3, we show the interaction between a set of genes that were selected after applying our rule for removing unimportant variables.

Finally, we show average runtimes in Table 3. In order to make a fair comparison, the runtime includes the time needed to produce the optimal precision matrix from a given input dataset. This includes not only the time to solve each optimization problem but also the time to compute the covariance matrix (if needed). Our Riccati method is considerably faster than the sparse method.

6 Concluding Remarks

We can generalize our penalizer to one of the form $\|\mathbf{A}\boldsymbol{\Omega}\|_{\mathfrak{F}}^2$ and investigate under which conditions, this problem has a low-rank solution. Another extension includes finding inverse covariance matrices that are both sparse and low-rank. While in this paper we show loss consistency, we could also analyze conditions for which the recovered parameters approximate the ground truth, similar to the work of (Rothman et al., 2008; Ravikumar et al., 2011). Proving consistency of sparse patterns is a challenging line of research, since our low-rank estimators would not have enough degrees of freedom in order to accomodate for all possible sparseness patterns. We conjecture that such results might be possible by borrowing from the literature on principal component analysis (Nadler, 2008).

References

Banerjee, O., El Ghaoui, L., & d'Aspremont, A. (2008). Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *JMLR*.

Banerjee, O., El Ghaoui, L., d'Aspremont, A., & Natsoulis, G. (2006). Convex optimization techniques for fitting sparse Gaussian graphical models. *ICML*.

Cohen, A., Dahmen, W., & DeVore, R. (2009). Compressed sensing and best k-term approximation. *J. Amer. Math. Soc.*

Dempster, A. (1972). Covariance selection. *Biometrics*.

Duchi, J., Gould, S., & Koller, D. (2008). Projected subgradient methods for learning sparse Gaussians. *UAI*.

Friedman, J., Hastie, T., & Tibshirani, R. (2007). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*.

Guillot, D., Rajaratnam, B., Rolfs, B., Maleki, A., & Wong, I. (2012). Iterative thresholding algorithm for sparse inverse covariance estimation. *NIPS*.

Hsieh, C., Dhillon, I., Ravikumar, P., & Banerjee, A. (2012). A divide-and-conquer procedure for sparse inverse covariance estimation. *NIPS*.

Hsieh, C., Sustik, M., Dhillon, I., & Ravikumar, P. (2011). Sparse inverse covariance matrix estimation using quadratic approximation. *NIPS*.

Huang, J., Liu, N., Pourahmadi, M., & Liu, L. (2006). Covariance matrix selection and estimation via penalized normal likelihood. *Biometrika*.

Johnson, C., Jalali, A., & Ravikumar, P. (2012). High-dimensional sparse inverse covariance estimation using greedy methods. *AISTATS*.

Johnstone, I. (2001). On the distribution of the largest principal component. *The Annals of Statistics*.

Lauritzen, S. (1996). *Graphical models*. Oxford Press.

Lim, Y. (2006). The matrix golden mean and its applications to Riccati matrix equations. *SIAM Journal on Matrix Analysis and Applications*.

Lu, Z. (2009). Smooth optimization approach for sparse covariance selection. *SIAM Journal on Optimization*.

Mazumder, R., & Hastie, T. (2012). Exact covariance thresholding into connected components for large-scale graphical lasso. *JMLR*.

Meinshausen, N., & Bühlmann, P. (2006). High dimensional graphs and variable selection with the lasso. *The Annals of Statistics*.

Nadler, B. (2008). Finite sample approximation results for principal component analysis: A matrix perturbation approach. *The Annals of Statistics*.

Ng, A. (2004). Feature selection, ℓ_1 vs. ℓ_2 regularization, and rotational invariance. *ICML*.

Olsen, P., Oztoprak, F., Nocedal, J., & Rennie, S. (2012). Newton-like methods for sparse inverse covariance estimation. *NIPS*.

Ravikumar, P., Wainwright, M., Raskutti, G., & Yu, B. (2011). High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Electronic Journal of Statistics*.

Rothman, A., Bickel, P., Levina, E., & Zhu, J. (2008). Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*.

Scheinberg, K., Ma, S., & Goldfarb, D. (2010). Sparse inverse covariance selection via alternating linearization methods. *NIPS*.

Scheinberg, K., & Rish, I. (2010). Learning sparse Gaussian Markov networks using a greedy coordinate ascent approach. *ECML*.

Schmidt, M., van den Berg, E., Friedlander, M., & Murphy, K. (2009). Optimizing costly functions with simple constraints: A limited-memory projected quasi-Newton algorithm. *AISTATS*.

Witten, D., & Tibshirani, R. (2009). Covariance-regularized regression and classification for high-dimensional problems. *Journal of the Royal Statistical Society*.

Yuan, M., & Lin, Y. (2007). Model selection and estimation in the Gaussian graphical model. *Biometrika*.

Yun, S., Tseng, P., & Toh, K. (2011). A block coordinate gradient descent method for regularized convex separable optimization and covariance selection. *Mathematical Programming*.

Zhou, S., Rütimann, P., Xu, M., & Bühlmann, P. (2011). High-dimensional covariance estimation based on Gaussian graphical models. *JMLR*.

Discovering Cyclic Causal Models with Latent Variables: A General SAT-Based Procedure

Antti Hyttinen **Patrik O. Hoyer**
HIIT & Dept. of Computer Science
University of Helsinki
Finland

Frederick Eberhardt
Philosophy
California Institute of Technology
Pasadena, CA, USA

Matti Järvisalo
HIIT & Dept. of Computer Science
University of Helsinki
Finland

Abstract

We present a very general approach to learning the structure of causal models based on d-separation constraints, obtained from any given set of overlapping passive observational or experimental data sets. The procedure allows for both directed cycles (feedback loops) and the presence of latent variables. Our approach is based on a logical representation of causal pathways, which permits the integration of quite general background knowledge, and inference is performed using a Boolean satisfiability (SAT) solver. The procedure is complete in that it exhausts the available information on whether any given edge can be determined to be present or absent, and returns "unknown" otherwise. Many existing constraint-based causal discovery algorithms can be seen as special cases, tailored to circumstances in which one or more restricting assumptions apply. Simulations illustrate the effect of these assumptions on discovery and how the present algorithm scales.

1 INTRODUCTION

One of the main goals in many fields of science is to identify the causal relations existing among some set of variables of interest. Such causal knowledge may be inferred both from experimental data (randomized controlled trials) and passive observational measurements. In general the information available from multiple such studies may need to be combined to obtain an accurate picture of the underlying system. In recent years, many approaches to this *causal discovery* problem have been suggested (Spirtes et al., 1999; Richardson and Spirtes, 1999; Schmidt and Murphy, 2009; Claassen and Heskes, 2010; Peters et al., 2010; Triantafillou et al., 2010), building on the framework of causal Bayes networks (Spirtes et al., 1993; Pearl, 2000). In this framework, causal relations among a set of variables $\mathbf{V}$ are represented by a directed graph $\mathcal{G}$ in which each variable is represented by a node in the graph, and an arrow from node x to node y indicates that x is a direct cause of y (with respect to $\mathbf{V}$).

Although causal models based on directed graphs are often restricted to be *acyclic*, causal feedback can be represented by permitting directed cycles in $\mathcal{G}$, i.e. directed paths from a node back to itself. In addition, unmeasured common causes of two or more nodes in $\mathbf{V}$ are commonly represented by allowing bi-directed arrows ($\leftrightarrow$) between any pair of confounded nodes.[1] (If there are no such confounders, the set $\mathbf{V}$ is said to be *causally sufficient*.) Thus, in the most general case of cyclic causal structures with latent variables, any pair of nodes $x, y \in \mathbf{V}$, with $x \neq y$, can be connected by any combination of the edges $x \rightarrow y$, $y \rightarrow x$, and $x \leftrightarrow y$ (see Figure 1 for examples).

One of the key theoretical concepts in causal models based on directed graphs is the notion of *d-separation*, due to Geiger et al. (1990). This is a graphical separation criterion that provides the structural counterpart to (conditional) independencies in the probability distribution generated by the model. D-separation is based on paths in the graph. Since a single pair of nodes can be connected by multiple edges, in our model space a *path* is defined as a sequence of consecutive edges in the graph, without any restrictions on the types or orientations of the edges involved.

Definition 1 (D-separation) *A path p is said to be d-separated (or blocked) by a set of nodes* $\mathbf{C}$ *if and only if (i) p contains a chain* $i \rightarrow m \rightarrow j$ *or a fork* $i \leftarrow m \rightarrow j$ *such that the middle node m is in* $\mathbf{C}$*, or (ii) p contains an inverted fork (or collider)* $i \rightarrow m \leftarrow j$

[1] In this representation a latent variable affecting more than two observed variables is represented by two-way confounders (bi-directed edges) between all pairs of nodes corresponding to the affected observed variables.

such that the middle node m is not in $\mathbf{C}$ and such that no descendant of m is in $\mathbf{C}$. A set $\mathbf{C}$ is said to d-separate x from y if and only if $\mathbf{C}$ blocks every path from x to y. (Pearl, 2000)

When applying Definition 1 to graphs with bi-directed edges such as in Figure 1(b), the bidirected edge $z \leftrightarrow w$ can be viewed as a latent structure $z \leftarrow l_{zw} \rightarrow w$.

In acyclic models, such as causal Bayes networks, if two nodes x and y are d-separated given a set $\mathbf{C}$ then the corresponding random variables are statistically independent when conditioning on $\mathbf{C}$ in the probability distribution generated by the model. If there are no statistical independencies in the distribution other than those implied by d-separation applied to the underlying graph, the distribution is said to be *faithful* to the graph. Thus, under an assumption of faithfulness causal discovery procedures can use the outcomes of statistical independence tests, applied to the observed data, to infer d-separation and hence structural properties of the underlying graph. For example, if in a set of four variables $\mathbf{V} = \{x, y, z, w\}$ it is found that (i) x is unconditionally independent of y, (ii) x is independent of w given z, (iii) y is independent of w given z, and (iv) no other unconditional independencies are found, then the well-known PC-algorithm (Spirtes et al., 1993) will infer that the underlying causal structure is the one in Figure 1(a).

While the correspondence between probabilistic independence and d-separation is known to hold generally for acyclic models (even when there are latent variables), the case is not as clear for cyclic models. The correspondence is known to hold for linear causal relations with Gaussian error terms, i.e. non-recursive linear Gaussian structural equation models (Spirtes, 1995), and can be extended to models with correlated error terms, which is one way to account for causally insufficient sets of variables. A general characterization of the parameterizations of cyclic models (with latent variables), for which the correspondence between d-separation and probabilistic independence holds, is not known (Pearl and Dechter, 1996; Neal, 2000).

Following the standard approach of non-parametric causal discovery algorithms, we use d-separation relations as the basic input to our procedure, but acknowledge that in the cyclic case only the linear Gaussian models are known to provide the appropriate correspondence with statistical independence. We allow for a set-up similar to the overlapping data sets approach of the ION-procedure (Tillman et al., 2009) in that we do not restrict ourselves to a single data set measured over some set of observed nodes, but can handle d-separation relations that were obtained from different (overlapping) sets $\mathbf{V}_i$ of nodes. Analogously to Hyttinen et al. (2012) we generalize the overlapping data sets case to allow that the $\mathbf{V}_i$ can contain nodes that are known to have been subject to a randomized experiment. Nevertheless, the target of our discovery procedure is the underlying causal graph $\mathcal{G}$ over the set of nodes $\mathbf{V} = \bigcup_i \mathbf{V}_i$, implying that $\mathcal{G}$ may contain edges between nodes that are never measured together in the same data set.

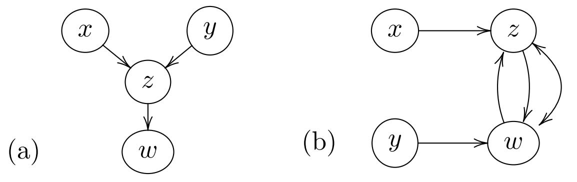


Figure 1: Example graphs (see text for details).

2 PROBLEM SETTING

We consider the space of cyclic causal models $\mathcal{G}$ over the (jointly) causally *in*sufficient set of nodes $\mathbf{V} = \bigcup_i \mathbf{V}_i$, where each $\mathbf{V}_i$ specifies the nodes present in experiment $\mathcal{E}_i = (\mathbf{J}_i, \mathbf{U}_i)$. $\mathbf{J}_i$ and $\mathbf{U}_i$ form a partition of $\mathbf{V}_i$ such that the nodes in $\mathbf{J}_i$ are randomized simultaneously and independently and the nodes in $\mathbf{U}_i$ are passively observed ($\mathbf{J}_i$ can be empty to allow for passive observational settings). We use the following simplification of the d-separation criterion:

Definition 2 (D-separation) *A path 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").*

Definition 2 is equivalent to Definition 1 when an edge can be used multiple times in a path (Studený, 1998; Koster, 2002). For example, the sequence of edges $x \rightarrow z \rightarrow w \leftarrow z \leftarrow y$ in the graph in Figure 1(a) is d-connecting with respect to conditioning set $\mathbf{C} = \{w\}$. The extension of d-separation to experimental settings is straightforward: a d-connecting path may only contain a node $x \in \mathbf{J}$ if $x \notin \mathbf{C}$ and x is a fork (common cause) on the path or the source of the path. We write $x \perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$ (resp. $x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$) to denote that x is d-separated from (resp. d-connected to) y given $\mathbf{C}$ in the experiment with intervention set $\mathbf{J}$. We assume we have a d-separation oracle that returns the truth values of statements of the form '$x \perp y \,|\, \mathbf{C} \,||\, \mathbf{J}_i$' in the true graph $\mathcal{G}$, for any pair of distinct nodes x, y and set of nodes $\mathbf{C}$ that occur together in some $\mathbf{V}_i$.

It is well known that even in the presence of randomized experiments the set of all d-separation relations over the set of nodes in general underdetermines the true causal structure even for much more restricted model spaces than we consider here. So the discovery task is to determine for each pair of nodes in $\mathbf{V}$ and for each edge type ($\leftarrow, \rightarrow, \leftrightarrow$) whether the edge is present, absent or if its existence is unknown. In addition we determine possible indirect (ancestral) causal relations: for each ordered pair of nodes (x, y), whether a directed path $x \rightarrow \cdots \rightarrow y$ exists, does not exist or if its existence is unknown.

3 SAT AND BACKBONES

Our algorithm for causal structure discovery is based on computing the so-called backbone of a given formula in propositional logic. We employ a Boolean satisfiability (SAT) solver (Biere et al., 2009) to determine the backbone, which can be directly interpreted as the solution to the structure discovery task. This section provides an overview on SAT and backbones.

Propositional formulas are built from Boolean variables by repeated application of the connectives $\neg$ (negation), $\vee$ (disjunction, logical OR), $\wedge$ (conjunction, logical AND), $\Rightarrow$ (implication) and $\Leftrightarrow$ (equivalence). Any propositional formula can be represented in *conjunctive normal form* (CNF) using a standard linear-size encoding (Tseitin, 1983). For a Boolean variable X, there are two literals, X and $\neg X$. A *clause* is a disjunction of literals; a CNF formula is a conjunction of clauses. A truth assignment is a function τ from Boolean variables to $\{0, 1\}$. A clause C is satisfied by τ if $\tau(X) = 1$ for some literal X in C, or $\tau(X) = 0$ for some literal $\neg X$ in C. A CNF formula F is satisfiable if there is an assignment that satisfies all clauses in F, and unsatisfiable otherwise. The NP-complete Boolean satisfiability (SAT) problem asks whether a given CNF formula F is satisfiable.

Implementations of decision procedures for SAT, so-called SAT solvers, can in practice not only determine satisfiability of CNF formulas, but also produce a satisfying truth assignment for satisfiable formulas. The most efficient SAT solvers are based on the complete conflict-driven clause learning (CDCL) search algorithm (Marques-Silva and Sakallah, 1999; Moskewicz et al., 2001; Eén and Sörensson, 2004). Central to CDCL is the ability to derive lemmas (in terms of new CNF clauses) based on non-solutions detected during search, which makes the search performed by CDCL SAT solvers differ from standard depth-first backtracking search. In many cases, the state-of-the-art CDCL SAT solvers can solve SAT instances consisting of millions of clauses and variables (Järvisalo et al., 2012).

If a Boolean variable X takes the same value in all satisfying truth assignments of a given CNF formula F, X is called a *backbone variable* of F; the value X is assigned to in all satisfying assignments is called the polarity of X. The set of backbone variables (or simply, the backbone) of a formula F can be computed by a linear number of calls (in the number of variables in F) to a SAT solver: if exactly one of $F \wedge X$ and $F \wedge \neg X$ is satisfiable, then X is in the backbone of F.

4 ENCODING D-SEPARATION

Figure 2 shows our propositional encoding for the d-connection property. The encoding allows to represent both d-separation and d-connection relations as constraints directly on the edges present or absent in the underlying causal graph. In essence, the encoding spells out Definition 2 (extended to experiments) by expressing the conditions for paths being blocked or unblocked.

In the encoding, Boolean variables $[x \rightarrow y]$ and $[x \leftrightarrow y]$ represent the underlying causal graph. For each pair of distinct nodes $x, y \in \mathbf{V}$, the Boolean variable $[x \rightarrow y]$ (variable $[x \leftrightarrow y]$, respectively) takes the value 1 if and only if the edge $x \rightarrow y$ (edge $x \leftrightarrow y$, respectively) is present in the graph.[2] The Boolean variable $[x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}]$ is 1 if and only if x and y are d-connected in the underlying graph when conditioning on $\mathbf{C}$ and intervening on $\mathbf{J}$. To encode the different types of d-connecting paths of length l between pairs of nodes x, y when conditioning on $\mathbf{C}$ and intervening on $\mathbf{J}$ (Eqs. 1–7), Boolean variables $[x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{-\cdots>}} y]$, $[x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{<\cdots>}} y]$, and $[x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{-\cdots-}} y]$ are introduced, with the respective arrowheads and edge-tails as indicated. In general, d-connecting paths in a cyclic graph can have infinite length, length of a path being the number of its edges. However, as shown in Appendix B, only paths of a maximum length $l_{\max} = 2|\mathbf{V}| - 4$ need to be considered. These Boolean variables are hence defined for all paths (of the four types) of length $l = 1, \ldots, l_{\max}$ and for all pairs of nodes in $\mathbf{V}$.

The constraint requiring that a specific d-connection $x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$ is present is constructed by taking the conjunction of the variable $[x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}]$ and Equations 1–7. Similarly, the constraint requiring that a specific d-separation $x \perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$ is present is the conjunction of $\neg[x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}]$ and Equations 1–7.

From a causal perspective, for a d-connection $x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$ the encoding splits the d-connecting paths into four groups (Eq. 1): (i) paths that start with an edge-tail at x and end with an arrowhead at y, (ii) paths that start with an arrowhead at x and end with an edge-tail at y, (iii) paths that start with an arrowhead at x and end with an arrowhead at y, and (iv)

[2]We omit self-loops, i.e. edges from a node to itself, as they do not affect the d-connectedness of a graph.

Encoding of d-connection between nodes x, y given conditioning set $\mathbf{C}$ and intervention set $\mathbf{J}$.

$$[x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}] \quad \Leftrightarrow \quad \bigvee_{l=1}^{l_{\max}} \left([x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} y] \vee [y \overset{l}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} x] \vee [x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{\dashleftarrow\!\dashrightarrow}} y] \vee [x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{-\cdots-}} y] \right) \tag{1}$$

Paths of length 1:

$$[x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} y] \quad \Leftrightarrow \quad \begin{cases} [x \to y] & \text{if } y \notin \mathbf{J} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$[x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashleftarrow\!\dashrightarrow}} y] \quad \Leftrightarrow \quad \begin{cases} [x \leftrightarrow y] & \text{if } x \notin \mathbf{J} \text{ and } y \notin \mathbf{J} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$[x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{-\cdots-}} y] \quad \Leftrightarrow \quad 0 \tag{4}$$

Paths of length $l = 2, \ldots, l_{\max}$:

$$[x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} y] \quad \Leftrightarrow \quad \bigvee_{z \notin \mathbf{C}} \left([x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} z] \wedge [z \overset{l-1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} y] \right) \vee \bigvee_{z \in \mathbf{C}} \left([x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} z] \wedge [z \overset{l-1}{\underset{\mathbf{C},\mathbf{J}}{\dashleftarrow\!\dashrightarrow}} y] \right) \tag{5}$$

$$\begin{aligned} [x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{\dashleftarrow\!\dashrightarrow}} y] \quad \Leftrightarrow \quad & \bigvee_{z \notin \mathbf{C}} \left([z \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} x] \wedge [z \overset{l-1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} y] \right) \vee \bigvee_{z \notin \mathbf{C}} \left([z \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} x] \wedge [z \overset{l-1}{\underset{\mathbf{C},\mathbf{J}}{\dashleftarrow\!\dashrightarrow}} y] \right) \vee \\ & \bigvee_{z \notin \mathbf{C}} \left([x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashleftarrow\!\dashrightarrow}} z] \wedge [z \overset{l-1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} y] \right) \vee \bigvee_{z \in \mathbf{C}} \left([x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashleftarrow\!\dashrightarrow}} z] \wedge [z \overset{l-1}{\underset{\mathbf{C},\mathbf{J}}{\dashleftarrow\!\dashrightarrow}} y] \right) \end{aligned} \tag{6}$$

$$[x \overset{l}{\underset{\mathbf{C},\mathbf{J}}{-\cdots-}} y] \quad \Leftrightarrow \quad \bigvee_{z \notin \mathbf{C}} \left([x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} z] \wedge [z \overset{l-1}{\underset{\mathbf{C},\mathbf{J}}{-\cdots-}} y] \right) \vee \bigvee_{z \in \mathbf{C}} \left([x \overset{1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} z] \wedge [y \overset{l-1}{\underset{\mathbf{C},\mathbf{J}}{\dashrightarrow}} z] \right) \tag{7}$$

Figure 2: Encoding d-connection via paths between pairs of nodes.

paths that start with an edge-tail at x and end with an edge-tail at y. The paths are built up recursively in terms of length l (Eqs. 5, 6, and 7). By keeping track of the path lengths we ensure that each path bases out through Eqs. 2 and 3 on the actual edges in the graph, whose presence is represented by Boolean variables $[x \to y]$ and $[x \leftrightarrow y]$. There are no paths of type (iv) with length 1, as such a path must involve at least one collider (in $\mathbf{C}$) to have tails at both ends (hence Eq. 4). The shortest valid case is of length $l = 2$ and results from the second half of Eq. 7. By explicitly keeping track of the terminal edge-marks in each path variable, the encoding ensures that all colliders on a d-connecting path are in the conditioning set $\mathbf{C}$, and all non-colliders are not in $\mathbf{C}$. The base cases (Eqs. 2 and 3) ensure that there is no path with an edge into a variable that is intervened on (into $y \in \mathbf{J}$).

For each given d-separation relation $x \perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$ (or similarly each d-connection relation $x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$), the whole encoding, including Eqs. 1–7, is cubic in the number $|\mathbf{V}|$ of nodes. Furthermore, it is important to notice that our algorithm, as described next, does not generate the constraints in Eqs. 1–7 for all possible d-separation and d-connection relations at once. The constraints for individual relations are generated only on demand during the execution of the algorithm, in many cases avoiding generating an exponential number of constraints needed to represent all possible d-separation and d-connection relations.

The SAT-based approach to causal structure discovery by Triantafillou et al. (2010) uses an encoding based on partial ancestral graphs (PAGs), a particular form of equivalence class. Their encoding does not suffice for our purposes, since it is restricted to *acyclic* causal structures in *non-experimental* settings, and given experiments it is often possible to distinguish between different graphs that for passive observational data belong to the same PAG.

5 ALGORITHM

The encoding of d-separation relations presented in the previous section can be used for a variety of discovery applications. For the purpose of illustration we will present here one algorithm for a common discovery setting. The extension to other settings is then easily explained. Algorithm 1 iterates over three steps until all d-separation relations are known: (1) finding a set of d-separation/connection tests T_c (in order of increasing conditioning set size) with currently unknown result, and determining those relations D_c, (2) generating the additional constraints encoding the relations in D_c (recall the encoding in Figure 2), and (3) computing the backbone over the propositional formula

Algorithm 1 SAT-based causal structure discovery

```
Initialize solution $S$ for the edge variables $[x \rightarrow y]$, $[x \leftrightarrow y]$
of each pair of distinct nodes $x, y \in \mathbf{V}$ to status *unknown*.

Initialize $\varphi$ to be the empty propositional formula.

For conditioning set size $c$ from $0$ to $|\mathbf{V}| - 2$:
  1: *Determine d-separation/connection relations.*
    Find a set $T_c$ of d-separation/connection tests with conditioning set size $c$ that are undetermined given $S$.
    Determine the d-separation/connection relation for each test in $T_c$, and let set $D_c$ consist of these relations.
  2: *Refine the working formula $\varphi$.*
    For each $x \perp y \,|\mathbf{C}\,||\mathbf{J}$ in $D_c$:
      *Encode $x \perp y \,|\mathbf{C}\,||\mathbf{J}$ using equations 1-7:*
      $\varphi := \varphi \wedge$ Encode($x \perp y \,|\mathbf{C}\,||\mathbf{J}$).
    For each $x \not\perp y \,|\mathbf{C}\,||\mathbf{J}$ in $D_c$:
      *Encode $x \not\perp y \,|\mathbf{C}\,||\mathbf{J}$ using equations 1-7:*
      $\varphi := \varphi \wedge$ Encode($x \not\perp y \,|\mathbf{C}\,||\mathbf{J}$).
  3: *Incremental backbone computation with SAT solver*
    Compute $B$: the set of edge-variables $[x \rightarrow y]$, $[x \leftrightarrow y]$ in the backbone of $\varphi$.
    For each edge variable $e$ in $B$:
      If $e \in B$ with polarity 1, set status of $e$ to *present* in $S$.
      If $e \in B$ with polarity 0, set status of $e$ to *absent* in $S$.

Output $S$: the status of each edge.
```

consisting of the constraints generated so far.

In Step 1 we apply a pruning heuristic (described in Appendix A) that guarantees that all unknown d-separation relations are found, but remains computationally tractable. We use a d-separation oracle (see Section 2) to determine the result of each test.

In Step 2, given a d-connection relation $x \not\perp y \,|\mathbf{C}\,||\mathbf{J}$, the subroutine Encode returns the conjunction of $[x \not\perp y \,|\mathbf{C}\,||\mathbf{J}]$ and the formulas in Eqs. 1–7. Similarly, given a d-separation relation $x \perp y \,|\mathbf{C}\,||\mathbf{J}$, Encode returns the conjunction of $\neg[x \not\perp y|\,\mathbf{C}||\,\mathbf{J}]$ and the formulas in Eqs. 1–7. Note that for each combination of $\mathbf{C}$ and $\mathbf{J}$, Eqs. 2–7 need to the added only once into φ (also guaranteed by our current implementation). This is important in practice, so that the SAT solver is not suffocated with many copies of the same constraints.

In Step 3, a SAT solver is used incrementally for determining which of the edge-variables in the current working formula φ are in the backbone of φ. The polarity of these backbone variables determines whether the corresponding edges are present or absent.

Like other constraint based causal discovery algorithms, Algorithm 1 considers d-separation relations in order of the size of the conditioning set $\mathbf{C}$. For sparse graphs, this enables a rapid pruning of the constraint generation on the basis of the simplest tests. But unlike other constraint based algorithms, Algorithm 1 can explicitly include known d-connections, rather than assuming that there is a d-connection whenever no d-separation is found (see also Section 7).

Algorithm 1 is easily adjusted to consider an *arbitrary set* of d-separation/connection relations as input, as long as the set of nodes $\mathbf{V}$ is specified from the outset. If the set is small, one can just run step 2 and 3 to compute the backbone using all available relations, otherwise one can run the full procedure, simply omitting relations from D_c that are not available in the set. It will terminate when all relations are encoded or when no more are needed, as determined by step 1.

In Algorithm 1 we use the status on each edge as the output. If other aspects of the graphs are of interest, one can easily define other variables and compute the backbone over them. In Section 6 we use this feature to determine which ancestral relationships are known.

5.1 BACKGROUND KNOWLEDGE AND MODEL SPACE ASSUMPTIONS

Although we have considered a very general model space, restricting the procedure to smaller spaces is simple. Focusing on just one data set rather than a set of overlapping data sets, or only considering passive observational data and no experiments, requires no adjustments of Algorithm 1. If one has reason to believe that there are no unmeasured nodes, i.e. that $\mathbf{V}$ is (jointly) causally sufficient, then setting

$$[x \leftrightarrow y] \Leftrightarrow 0 \tag{8}$$

for all pairs of nodes in the encoding will enforce this restriction. If one is only interested in acyclic causal structures, then adding the constraint

$$\neg[x \not\perp y \,|\emptyset\,||\{x\}] \vee \neg[x \not\perp y \,|\emptyset\,||\{y\}] \tag{9}$$

for each pair of nodes, together with the respective path definitions (Eqs. 2–7), is sufficient. Eq. 9 disallows cycles by enforcing that there cannot be a directed path from x to y *and* a directed path from y to x. Since the conditioning set in each of the d-connection claims in Eq. 9 is empty, there cannot exist any colliders in the d-connecting paths. The intervention on x and y, respectively, in each of the claims in Eq. 9 ensures that d-connections due to common causes are excluded. Only *directed* paths are involved in $x \not\perp y \,|\emptyset\,||\{x\}$ and $x \not\perp y \,|\emptyset\,||\{y\}$. In Section 5.2 we use this flexibility to generate the same causal inferences as other d-separation based algorithms.

More generally, the encoding allows for including various types of background knowledge. One can enforce that a particular edge is present or absent, that particular ancestral relations are maintained or disallowed, that specific paths (with, if needed, particular way-points and of a specific length) are present or absent. The type of knowledge that can be encoded is more general than any other constraint based procedure we are aware of, including the additions to the cSAT+ algorithm by Borboudakis et al. (2011). One is in principle only limited by what can be encoded in terms of a Boolean constraint over the edge and path variables. We think this could be of enormous utility to applications with significant domain knowledge or when qualitative causal relations are discovered by other means (e.g. using the additive noise or non-Gaussian techniques of Peters et al. (2010) and Hoyer et al. (2008)).

5.2 COMPLETENESS

For more restricted model spaces, graphical representations of the classes of d-separation-equivalent graphs have been developed (e.g. partial ancestral graphs). We do not have a similar representation for our more general model space and it is unclear whether an easily interpretable representation is possible, since there can be graphs that share the same d-separation relations, but differ in adjacencies, orientations *and* ancestral relationships. By only providing the status of each edge as output of Algorithm 1, we follow Triantafillou et al. (2010) who used this solution format in light of the often (even computationally) unmanageable output of the ION-algorithm (which does not consider cyclic graphs). The downside is that this output is not fully informative about the solution space. For example, if d-separation relations were obtained from a passive observation of the graph $x \to y \to z$, then the current output does not represent that $x \to y \leftarrow z$ is not a solution. Instead, it would (among other things) mark all edges of adjacent nodes as unknown, since $x \leftarrow y \leftarrow z$ is also a solution. Nevertheless, it is trivial to query our procedure about graphical aspects that are not represented in the output. Since the complete solution space is implicitly represented by the working formula φ, the SAT-solver can easily determine that $x \to y \leftarrow z$ is not a valid solution in this example. Similarly, one can query the status of any other structural proposition by constructing a Boolean variable X for it using the edge or path variables in the encoding, and determining whether X is in the backbone of φ or not. If it is, then polarity 1 indicates that X is true for *all* graphs that satisfy φ, while polarity 0 indicates that X is false for *all* graphs that satisfy φ. If X is not in the backbone, then there is a graph $\mathcal{G}_1$ that satisfies φ, for which X is true, and a graph $\mathcal{G}_2$ that satisfies φ, but for which X is false. This is one, given the encoding perhaps trivial, sense in which our procedure is complete for any propositional query given the d-separation/connection relations (and any model space restrictions) as input. We call this *query-completeness.*

A different type of completeness is used in the context of other constraint based algorithms. Given the d-separation tests that an algorithm performs, we say that an algorithm is *d-separation complete* if all d-separation relations over the set of nodes are known. The PC-algorithm (for acyclic graphs over a causally sufficient set of nodes), the FCI-algorithm (for acyclic graphs over a causally *in*sufficient set of nodes) and the CCD-algorithm (for *cyclic* graphs over a causally sufficient set of nodes) are all d-separation complete for their model spaces, respectively (Spirtes et al., 1993; Richardson, 1996; Spirtes et al., 1999). Relying on the model space assumptions, the algorithms conduct just enough d-separation tests to determine *all* d-separation relations of the graphs in the solution space, even relations that the algorithms did not explicitly test. None of these algorithms are d-separation complete when their model space assumptions are violated: Figure 1(b) gives a cyclic graph for which FCI is not d-separation complete, since it does not test whether $x \perp y \,|\{w, z\}$. The graph with latent confounders in Figure 3 is an example for which CCD is not d-separation complete, because it does not determine the d-separation $x_1 \perp x_5 \,|\{x_2, x_3, x_4\}$. PC does not handle either graph. These limitations illustrate that achieving d-separation completeness without performing all tests is a non-trivial problem in the general model space we consider (containing both graphs). Once we consider overlapping data sets, there are d-separation relations involving nodes that do not occur together in any $\mathbf{V}_i$. Sometimes these can be determined from the other d-separation relations, but often they remain undetermined even when all the d-separation relations within each $\mathbf{V}_i$ are established. For this setting we adjust the definition of d-separation completeness to require that exactly those relations that cannot be determined (in the sense just described) are left *unknown* and all others are determined.

For cyclic models with latent variables in overlapping experimental or observational data sets, Algorithm 1 *is* d-separation complete, and in general it will not test all available d-separation relations. But in the present implementation (of step 1) we resort to simple safe heuristics to avoid some redundant tests, and otherwise apply brute force (see Appendix A). It is an open challenge to further reduce the number of tests performed while preserving d-separation completeness. We cannot employ a simple variant of the efficient test schedules of FCI and CCD, as they select subsequent tests on the basis of a graphical representation of the knowledge acquired so far that is specific to their restricted model spaces. But *given* those restrictions, we can adopt the test schedules: Using FCI as basis, the ION-algorithm (which also assumes acyclicity) is d-separation complete for pas-

$$\begin{tikzcd}
x_1 \arrow[r, leftrightarrow] & x_2 \arrow[rrr, bend left] & x_3 \arrow[l] \arrow[r] & x_4 \arrow[r, leftrightarrow] \arrow[lll, bend right] & x_5
\end{tikzcd}$$

Figure 3: An (acyclic) graph with latents for which the CCD-algorithm is not d-separation complete.

sively observed overlapping data sets (Tillman et al., 2009). Similarly, if we assume acyclicity, we can use instead of our heuristic the test schedule of FCI in Algorithm 1 when analyzing overlapping *experimental* data sets: run FCI on each individual data set (experimental or not) and input to Algorithm 1 the results of the tests that FCI considered on each individual data set (together with the acyclicity constraints in (9) for the FCI model space). Algorithm 1 will then combine the information across data sets and output all information available on the status of each edge in the true graph. The set of d-separation relations tested by FCI is sufficient for d-separation completeness for the $\mathbf{V}_i$ in that data set. Interventions do not affect the d-separation completeness, since the manipulated graph in any experiment still satisfies all model space assumptions of FCI. (One could avoid some tests by further book-keeping of the information about the interventions, but for d-separation completeness it is unnecessary.) Given FCI's d-separation completeness on each data set, the constraints generated by feeding the test results to Algorithm 1 imply that all d-separations relations that could be tested, are already determined. Any d-separation relation still unknown cannot be determined. By assuming acyclicity we thus obtain d-separation completeness *using the efficient FCI schedule of tests* for overlapping data sets with experiments. As Algorithm 1 is also query-complete, we have a general procedure for the approaches of Lagani et al. (2012).

An analogous argument for *cyclic* graphs *without* latent nodes, using the test schedule of CCD, can only be made if we assume that the nodes in $\mathbf{V}$ are all observed in all (possibly experimental) data sets. In the overlapping setting, causal sufficiency can be violated in the individual data sets and, as shown above, CCD is not d-separation complete for such a model space.

6 SIMULATIONS

To determine the effectiveness of the proposed approach, we implemented Algorithm 1 and investigated the properties of the method empirically. Our implementation is based on the MiniSAT solver (Eén and Sörensson, 2004, 2003). The code is available at http://www.cs.helsinki.fi/group/neuroinf/nonparam/.

First, we investigated the extent to which our approach, and in particular the SAT solver used, is able to solve the large problem instances generated by non-trivial graphs. We generated random directed graphs of size $n = 5 \ldots 12$ nodes, in which each of the edges (both directed and bidirected) was independently included with probability 0.2. We then generated a random set of 10 overlapping experiments, in each of which each node was independently and with equal probability chosen to be either intervened, passively observed, or unobserved. Finally, we computed all observable d-separation/connection relations; these constituted the input to our procedure.

Figure 4(a) gives, for each value of n, the median run-time based on 100 random problem instances, for the complete procedure (solid curve), and when only considering conditioning sets $\mathbf{C}$ with two or fewer elements (dotted curve). Note that most instances involving a relatively small number of nodes (on the order of 10 or less) can be solved by the complete procedure in minutes, if not seconds. We emphasize that these are not trivial problems: No other existing causal discovery procedure can handle our model space (allowing both latents and cycles), nor our very general experimental setup (overlapping data sets including interventions). At the same time, it is quite clear that, at least in its current implementation, the method does not scale to much larger numbers of variables. Scalability could be achieved with a more efficient search for unknown d-separations in Step 1 of the algorithm.

An effective way to reduce the run-time of the algorithm is to limit the size of the conditioning sets considered (dotted line in Figure 4(a)). While this means that completeness is not guaranteed, Figure 4(b) shows that in most cases very little is lost in terms of identifiability. We randomly sampled 100 problem instances as above, except that we now fixed the number of nodes to $n = 8$. The red solid curve shows the proportion of true directed edges (i.e. $x \to y$ in the true graph) which were identified as a direct edge (as opposed to unknown, since no errors are made). Similarly, the red dashed curve shows the identification of absences of direct edges, and the remaining curves indicate the amount of bidirected edges and existence of directed paths (ancestral relationships) identified. A key observation is that tests of higher order (roughly $|\mathbf{C}| \geq 3$) provide very little additional information over those involving smaller conditioning sets.

Finally, we investigated the extent to which our very general model space (allowing both cycles and latents) is detrimental to identification when the true model is more restricted. We generated a total of 300 random problem instances using the same procedure as above, each with $n = 8$ nodes, where the first 100 models were restricted to being acyclic, the second 100 were restricted to contain no latents (i.e. no edges of the form $x \leftrightarrow y$ in the true graph over $\mathbf{V}$), and the remaining

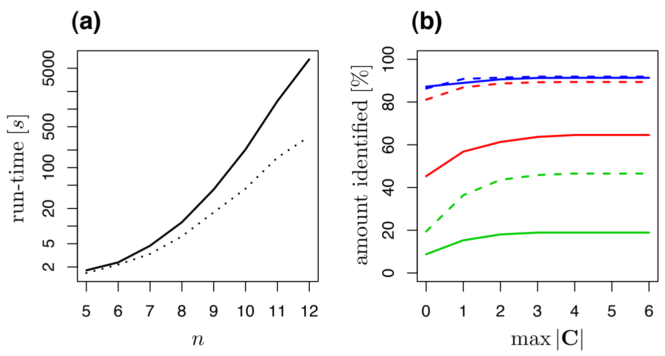


Figure 4: **(a)** Median run-time of the procedure as a function of the total number of nodes in the model. The dotted line gives the median run-time when restricting to max $|\mathbf{C}| = 2$. **(b)** Proportion of edges (solid lines) and absences of edges (dashed lines) identified, as a function of max $|\mathbf{C}|$. Directed edges are shown in red, bidirected edges (confounders) in green, and directed paths (ancestral relationships) in blue.

100 were both acyclic and contained no latents. Figure 5 shows the proportion of direct edges identified, and the proportion of absences of direct edges identified, as a function of the assumptions used (assuming an acyclic model, assuming no latents, assuming both, or assuming neither). The general message is that very little identifiability seems to be lost when assuming the more general model spaces in this experimental setup.

7 DISCUSSION

By focusing exclusively on d-separation and d-connection relations obtained *without errors* we have so far taken the approach used by other constraint-based algorithms in the literature (PC, FCI, CCD, ION, IOD, cSAT+ etc.) to separate the causal from the statistical inference. As an important direction for future work, we now briefly discuss integrating statistical inference.

In most realistic situations d-separation/connection relations are determined by independence tests from statistical data. Such tests, especially when performed in large numbers, produce errors due to the finite number of samples available and problems of multiple testing. All other constraint-based causal discovery algorithms face similar problems. In our case, the errors can result in d-separation/connection relations that are contradictory. Since the logical encoding is simply unsatisfiable in such cases, no output is given. But there are more interesting features of the encoding and the algorithm that hold promise to be useful with actual statistical data. First, since no definite answer is required of a d-separation test, we can enforce different p-value thresholds to detect independencies and dependencies (see Tsamardinos et al. (2012)). If

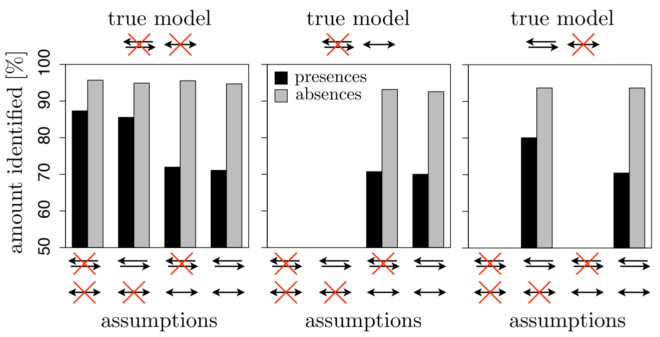


Figure 5: Proportion of directed edge presences and absences identified, under various model space assumptions, for acyclic true models without latents (left), acyclic models with latents (center), and cyclic models without latents (right).

a p-value of a test falls between the thresholds, the d-separation relation can be treated as unknown, by just not adding any constraints into the working formula φ. This approach does not completely avoid conflicts, but reduces their number and allows for at least some more control than many extant algorithms are able to offer. A second approach to dealing with statistical issues would be to exploit extensions of SAT, especially Boolean optimization in terms of maximum satisfiability (MaxSAT) of propositional formulas (Biere et al., 2009), where the task is to find a truth assignment that satisfies the maximum number of CNF clauses. Hence a MaxSAT solver could be used for discovering causal models that entail a minimal number of contradictory d-separation/connection relations in the input.

8 CONCLUSION

We presented a causal discovery procedure for a very general model space: to our knowledge, it is currently the only nonparametric causal discovery algorithm that allows for a model space that includes graphs with cycles *and* latent confounders (recall the discussion on cycles and d-separation in Section 1). The algorithm can be applied to overlapping data sets, whether they are experimental or passive observational, and can incorporate a large variety of different background information if available. It does not depend on parametric restrictions such as linearity (Hyttinen et al., 2012), and requires only the ability to test for d-separation/connection relations.

SAT-based procedures have been previously proposed for the more restricted space of acyclic causal models (Triantafillou et al., 2010; Borboudakis and Tsamardinos, 2012). However, ours is the first procedure that is complete with respect to overlapping surgical experiments, and additionally handles a model space that allows for cycles. In order to capture the more general model space, we employ a novel logical encoding of d-separation and d-connection relations. The Boolean constraints for individual relations are generated iteratively and only on demand during the execution of our procedure, and an incremental SAT solver is used for iteratively computing the backbone of the Boolean constraints. Our procedure can also be easily used for the more restricted model spaces by introducing additional Boolean constraints. By constraining the model space to causally sufficient or acyclic causal structures we can perform the inferences of the standard algorithms in the literature, such as PC, FCI, ION, IOD, cSAT+ and CCD for moderately sized graphs. The inferences made are complete in the most general and in the more restricted settings.

A PRUNING HEURISTICS

In the (intermediary) solution S describing our current knowledge some edges are present, some are absent and the presence of some edges is unknown. We consider two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, such that they agree on all the edges that are determined, but $\mathcal{G}_1$ omits all undetermined edges, while $\mathcal{G}_2$ includes all undetermined edges as present. As removing edges can only result in more d-separation relations, a d-connection relation present in $\mathcal{G}_1$ must be present in all solutions. Similarly, a d-separation relation present in $\mathcal{G}_2$ must be present in all possible solutions. Only the remaining tests are possibly informative. This is a safe heuristic that turns out to be computationally feasible, as forward calculation of d-separation/connection relations for a fully defined graph is fast for the model sizes we are considering. In addition, we also omit tests with conditioning sets that contain nodes that cannot be on a d-connecting path between the nodes in question.

B LIMIT ON THE PATH LENGTH

Written solely in terms of edge variables, the right-hand side of Eq. 1 is a large disjunction of d-connecting paths up to length $l_{\max}$ for the relation on the left-hand side. As a path of arbitrary length can be d-connecting, $l_{\max}$ should be infinite to guarantee soundness of the formulation. Here we show that only paths of lengths up to a certain upper bound need to be considered. The following lemma, proven at the end of this appendix, is essential in showing this.

Lemma 1 *If there exists a path that is d-connecting with respect to* $x \not\perp\!\!\!\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$ *and longer than* $2|\mathbf{V}| - |\mathbf{C} \cup \mathbf{J} \cup \{x, y\}| - 1$ *edges, then there exists a shorter path that is d-connecting with respect to the same relation.*

Consider a path p_{long} that is d-connecting for $x \not\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$ and longer than $2|\mathbf{V}| - |\mathbf{C} \cup \mathbf{J} \cup \{x, y\}| - 1$. By Lemma 1 there is a path p_{short} with at most length $2|\mathbf{V}| - |\mathbf{C} \cup \mathbf{J} \cup \{x, y\}| - 1$ edges that is d-connecting with respect to the same relation. Now the expanded version of the right hand side of Equation 1 has the form: $\ldots \vee [p_{\text{short}}] \vee [p_{\text{long}}] \vee \ldots$. The only situation where such a constraint may have a different value than $\ldots \vee [p_{\text{short}}] \vee \ldots$ is when p_{long} exists and p_{short} does not. This is impossible by the construction of p_{short} using Lemma 1. We can thus ignore $[p_{\text{long}}]$ and all paths longer than $2|\mathbf{V}| - |\mathbf{C} \cup \mathbf{J} \cup \{x, y\}| - 1$. We can set $l_{\max} = 2|\mathbf{V}| - 4$, since if $\mathbf{C} = \emptyset$, then paths have at most length $|\mathbf{V}| - 1$.

Proof of Lemma 1 The following six rules always give a shorter d-connecting path with respect to the same relation. The deleted part is underlined on the left. Circles indicate arrowhead or tail.

$$\underline{x\circ\cdots\circ} x\circ\cdots\circ y \quad \Rightarrow \quad x\circ\cdots\circ y \tag{10}$$

$$x\circ\cdots\circ y\underline{\circ\cdots\circ y} \quad \Rightarrow \quad x\circ\cdots\circ y \tag{11}$$

$$\cdots > \underline{z < \cdots >} z < \cdots \quad \Rightarrow \quad \cdots > z < \cdots \tag{12}$$

$$\cdots > \underline{z - \cdots \circ} z - \cdots \quad \Rightarrow \quad \cdots > z - \cdots \tag{13}$$

$$\cdots - \underline{z \circ \cdots -} z < \cdots \quad \Rightarrow \quad \cdots - z < \cdots \tag{14}$$

$$\cdots - \underline{z \circ \cdots \circ} z - \cdots \quad \Rightarrow \quad \cdots - z - \cdots \tag{15}$$

The rules imply that if a middle node z appears three times on a d-connecting path, the path will necessarily have at least one of the forms on the left in (12-15). (A path can never be d-connecting if the same node appears both as a collider and a non-collider somewhere on the path.) Thus a node can appear at most two times in paths that cannot be shortened.

First, consider the case with no colliders on the path. The only situation where a d-connecting path cannot be shortened and a node appears twice, occurs when the path has the form $\cdots > z - \cdots - z < \cdots$. This path cannot be d-connecting without a collider between the instances of z. Thus, without colliders a path that cannot be shortened has at most $|\mathbf{V}|$ nodes and thus length $|\mathbf{V}| - 1$. Second, if the path cannot be shortened, each node in $\mathbf{C} \cup \mathbf{J} \cup \{x, y\}$ can appear at most once due to (10-15). The remaining nodes can appear at most twice. This makes a total of $2|\mathbf{V}| - |\mathbf{C} \cup \mathbf{J} \cup \{x, y\}|$ nodes. Hence the length of the path is at most $2|\mathbf{V}| - |\mathbf{C} \cup \mathbf{J} \cup \{x, y\}| - 1$. $\square$

Acknowledgements

This research was supported by the Academy of Finland under grants 218147 and 255625 (POH), 132812 and 251170 (MJ), by HIIT (AH) and by the James S. McDonnell Foundation (FE).

References

Biere, A., Heule, M. J. H., van Maaren, H., and Walsh, T., editors (2009). *Handbook of Satisfiability.* IOS Press.

Borboudakis, G., Triantafilou, S., Lagani, V., and Tsamardinos, I. (2011). A constraint-based approach to incorporate prior knowledge in causal models. In *Proc. ESANN*, pages 321–326.

Borboudakis, G. and Tsamardinos, I. (2012). Incorporating causal prior knowledge as path-constraints in bayesian networks and maximal ancestral graphs. In *Proc. ICML*, pages 1799–1806.

Claassen, T. and Heskes, T. (2010). Causal discovery discovery in multiple models from different experiments. In *Proc. NIPS*, pages 415–423.

Eén, N. and Sörensson, N. (2003). Temporal induction by incremental SAT solving. *Electr. Notes Theor. Comput. Sci.*, 89(4):543–560.

Eén, N. and Sörensson, N. (2004). An extensible SAT-solver. In *Proc. SAT 2003*, pages 502–518.

Geiger, D., Verma, T., and Pearl, J. (1990). Identifying independence in Bayesian networks. *Networks*, 20:507–533.

Hoyer, P. O., Shimizu, S., Kerminen, A. J., and Palviainen, M. (2008). Estimation of causal effects using linear non-Gaussian causal models with hidden variables. *International Journal of Approximate Reasoning*, 49:362–378.

Hyttinen, A., Eberhardt, F., and Hoyer, P. O. (2012). Causal discovery of linear cyclic models from multiple experimental data sets with overlapping variables. In *Proc. UAI*, pages 387–396.

Järvisalo, M., Le Berre, D., Roussel, O., and Simon, L. (2012). The international SAT solver competitions. *AI Magazine*, 33(1):89–92.

Koster, J. T. A. (2002). Marginalizing and conditioning in graphical models. *Bernoulli*, 8(6):817–840.

Lagani, V., Tsamardinos, I., and Triantafilou, S. (2012). Learning from mixture of experimental data: A constraint-based approach. In *Proc. SETN*, pages 124–131.

Marques-Silva, J. P. and Sakallah, K. A. (1999). GRASP: A search algorithm for propositional satisfiability. *IEEE Trans. Computers*, 48(5):506–521.

Moskewicz, M. W., Madigan, C. F., Zhao, Y., Zhang, L., and Malik, S. (2001). Chaff: Engineering an efficient SAT solver. In *Proc. DAC*, pages 530–535.

Neal, R. (2000). On deducing conditional independence from d-separation in causal graphs with feedback. *Journal of Artificial Intelligence Research*, 12:87–91.

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 *Proc. UAI*, pages 420–426.

Peters, J., Janzing, D., and Schölkopf, B. (2010). Identifying cause and effect on discrete data using additive noise models. In *Proc. AISTATS*, pages 597–604.

Richardson, T. and Spirtes, P. (1999). Automated discovery of linear feedback models. In Glymour, C. and Cooper, G. F., editors, *Computation, Causation & Discovery*, pages 253–302. AAAI / MIT Press.

Richardson, T. S. (1996). *Feedback Models: Interpretation and Discovery.* PhD thesis, Carnegie Mellon University.

Schmidt, M. and Murphy, K. (2009). Modeling discrete interventional data using directed cyclic graphical models. In *Proc. UAI*, pages 487–495.

Spirtes, P. (1995). Directed cyclic graphical representation of feedback models. In *Proc. UAI*, pages 491–498.

Spirtes, P., Glymour, C., and Scheines, R. (1993). *Causation, Prediction, and Search.* Springer-Verlag.

Spirtes, P., Meek, C., and Richardson, T. (1999). An algorithm for causal inference in the presence of latent variables and selection bias. In Glymour, C. and Cooper, G. F., editors, *Computation, Causation & Discovery*, pages 211–252. AAAI / MIT Press.

Studený, M. (1998). Bayesian networks from the point of view of chain graphs. In *Proc. UAI*, pages 496–503.

Tillman, R. E., Danks, D., and Glymour, C. (2009). Integrating locally learned causal structures with overlapping variables. In *Proc. NIPS 2008*, pages 1665–1672.

Triantafillou, S., Tsamardinos, I., and Tollis, I. G. (2010). Learning causal structure from overlapping variable sets. In *Proc. AISTATS*, pages 860–867.

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.

Tseitin, G. S. (1983). On the complexity of derivation in propositional calculus. In *Automation of Reasoning 2: Classical Papers on Computational Logic 1967–1970*, pages 466–483. Springer.

Warped Mixtures for Nonparametric Cluster Shapes

Tomoharu Iwata
University of Cambridge
iwata.tomoharu@lab.ntt.co.jp

David Duvenaud
University of Cambridge
dkd23@cam.ac.uk

Zoubin Ghahramani
University of Cambridge
zoubin@eng.cam.ac.uk

Abstract

A mixture of Gaussians fit to a single curved or heavy-tailed cluster will report that the data contains many clusters. To produce more appropriate clusterings, we introduce a model which warps a latent mixture of Gaussians to produce nonparametric cluster shapes. The possibly low-dimensional latent mixture model allows us to summarize the properties of the high-dimensional clusters (or density manifolds) describing the data. The number of manifolds, as well as the shape and dimension of each manifold is automatically inferred. We derive a simple inference scheme for this model which analytically integrates out both the mixture parameters and the warping function. We show that our model is effective for density estimation, performs better than infinite Gaussian mixture models at recovering the true number of clusters, and produces interpretable summaries of high-dimensional datasets.

1 Introduction

Probabilistic mixture models are often used for clustering. However, if the mixture components are parametric (e.g. Gaussian), then the clustering obtained can be heavily dependent on how well each actual cluster can be modeled by a Gaussian. For example, a heavy tailed or curved cluster may need many components to model it. Thus, although mixture models are widely used for probabilistic clustering, their assumptions are generally inappropriate if the primary goal is to discover clusters in data. Dirichlet process mixture models can alleviate the problem of an unknown number of clusters, but this does not address the problem that real clusters may not be well matched by any parametric density.

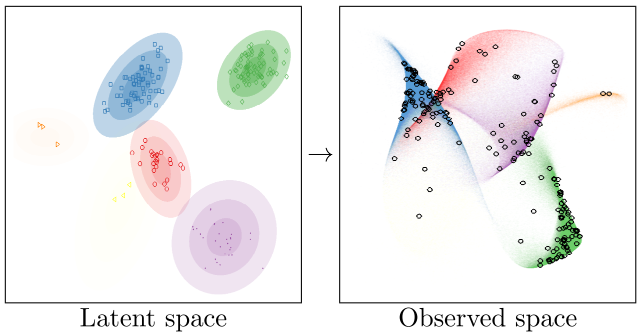


Figure 1: A sample from the iWMM prior. Left: In the latent space, a mixture distribution is sampled from a Dirichlet process mixture of Gaussians. Right: The latent mixture is smoothly warped to produce non-Gaussian manifolds in the observed space.

In this paper, we propose a nonparametric Bayesian model that can find nonlinearly separable clusters with complex shapes. The proposed model assumes that each observation has coordinates in a latent space, and is generated by warping the latent coordinates via a nonlinear function from the latent space to the observed space. By this warping, complex shapes in the observed space can be modeled by simpler shapes in the latent space. In the latent space, we assume an infinite Gaussian mixture model [1], which allows us to automatically infer the number of clusters. For the prior on the nonlinear mapping function, we use Gaussian processes [2], which enable us to flexibly infer the nonlinear warping function from the data. We call the proposed model the *infinite warped mixture model* (iWMM). Figure 1 shows a set of manifolds and datapoints sampled from the prior defined by this model.

To our knowledge this is the first probabilistic generative model for clustering with flexible nonparametric component densities. Since the proposed model is generative, it can be used for density estimation as well as clustering. It can also be extended to handle missing data, integrate with other probabilistic models, and use other families of distributions for the latent components.

We derive an inference procedure for the iWMM based on Markov chain Monte Carlo (MCMC). In particular, we sample the cluster assignments using Gibbs sampling, sample the latent coordinates using hybrid Monte Carlo, and analytically integrate out both the mixture parameters (weights, means and covariance matrices), and the nonlinear warping function.

2 Gaussian Process Latent Variable Model

In this section, we give a brief introduction to the Gaussian process latent variable model (GPLVM) [3], which can be viewed as a special case of the iWMM. The GPLVM is a probabilistic model of nonlinear manifolds. While not typically thought of as a density model, the GPLVM does in fact define a posterior density over observations [4]. It does this by smoothly warping a single, isotropic Gaussian density in the latent space into a more complicated distribution in the observed space.

Suppose that we have a set of observations $\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_N)^\top$, where $\mathbf{y}_n \in \mathbb{R}^D$, and they are associated with a set of latent coordinates $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_N)^\top$, where $\mathbf{x}_n \in \mathbb{R}^Q$. The GPLVM assumes that observations are generated by mapping the latent coordinates through a set of smooth functions, over which Gaussian process priors are placed. Under the GPLVM, the probability of observations given the latent coordinates, integrating out the mapping functions, is

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = (2\pi)^{-\frac{DN}{2}} |\mathbf{K}|^{-\frac{D}{2}} \exp\left(-\frac{1}{2}\mathrm{tr}(\mathbf{Y}^\top \mathbf{K}^{-1}\mathbf{Y})\right), \tag{1}$$

where $\mathbf{K}$ is the $N \times N$ covariance matrix defined by the kernel function $k(\mathbf{x}_n, \mathbf{x}_m)$, and $\boldsymbol{\theta}$ is the kernel hyperparameter vector. In this paper, we use an RBF kernel with an additive noise term:

$$k(\mathbf{x}_n, \mathbf{x}_m) = \alpha \exp\left(-\frac{1}{2\ell^2}(\mathbf{x}_n - \mathbf{x}_m)^\top(\mathbf{x}_n - \mathbf{x}_m)\right) + \delta_{nm}\beta^{-1}. \tag{2}$$

This likelihood is simply the product of D independent Gaussian process likelihoods, one for each output dimension.

Typically, the GPLVM is used for dimensionality reduction or visualization, and the latent coordinates are determined by maximizing the posterior probability of the latent coordinates, while integrating out the warping function. In that setting, the Gaussian prior density on $\mathbf{x}$ is essentially a regularizer which keeps the latent coordinates from spreading arbitrarily far apart. In contrast, we instead integrate out the latent coordinates as well as the warping function, and place a more flexible parameterization on $p(\mathbf{x})$ than a single isotropic Gaussian.

Just as the GPLVM can be viewed as a manifold learning algorithm, the iWMM can be viewed as learning a set of manifolds, one for each cluster.

3 Infinite Warped Mixture Model

In this section, we define in detail the infinite warped mixture model (iWMM). In the same way as the GPLVM, the iWMM assumes a set of latent coordinates and a smooth, nonlinear mapping from the latent space to the observed space. In addition, the iWMM assumes that the latent coordinates are generated from a Dirichlet process mixture model. In particular, we use the following infinite Gaussian mixture model,

$$p(\mathbf{x}|\{\lambda_c, \boldsymbol{\mu}_c, \mathbf{R}_c\}) = \sum_{c=1}^{\infty} \lambda_c \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c, \mathbf{R}_c^{-1}), \tag{3}$$

where λ_c, $\boldsymbol{\mu}_c$ and $\mathbf{R}_c$ is the mixture weight, mean, and precision matrix of the c^{th} mixture component. We place Gaussian-Wishart priors on the Gaussian parameters $\{\boldsymbol{\mu}_c, \mathbf{R}_c\}$,

$$p(\boldsymbol{\mu}_c, \mathbf{R}_c) = \mathcal{N}(\boldsymbol{\mu}_c|\mathbf{u}, (r\mathbf{R}_c)^{-1})\mathcal{W}(\mathbf{R}_c|\mathbf{S}^{-1}, \nu), \tag{4}$$

where $\mathbf{u}$ is the mean of $\boldsymbol{\mu}_c$, r is the relative precision of $\boldsymbol{\mu}_c$, $\mathbf{S}^{-1}$ is the scale matrix for $\mathbf{R}_c$, and ν is the number of degrees of freedom for $\mathbf{R}_c$. The Wishart distribution is defined as follows:

$$\mathcal{W}(\mathbf{R}|\mathbf{S}^{-1}, \nu) = \frac{1}{G}|\mathbf{R}|^{\frac{\nu-Q-1}{2}} \exp\left(-\frac{1}{2}\mathrm{tr}(\mathbf{S}\mathbf{R})\right), \tag{5}$$

where G is the normalizing constant. Because we use conjugate Gaussian-Wishart priors for the parameters of the Gaussian mixture components, we can analytically integrate out those parameters, given the assignments of points to components. Let z_n be the latent assignment of the n^{th} point. The probability of latent coordinates $\mathbf{X}$ given latent assignments $\mathbf{Z} = (z_1, \cdots, z_N)$ is obtained by integrating out the Gaussian parameters $\{\boldsymbol{\mu}_c, \mathbf{R}_c\}$ as follows:

$$p(\mathbf{X}|\mathbf{Z}, \mathbf{S}, \nu, r) = \prod_{c=1}^{\infty} \pi^{-\frac{N_c Q}{2}} \frac{r^{Q/2}|\mathbf{S}|^{\nu/2}}{r_c^{Q/2}|\mathbf{S}_c|^{\nu_c/2}} \times \prod_{q=1}^{Q} \frac{\Gamma(\frac{\nu_c+1-q}{2})}{\Gamma(\frac{\nu+1-q}{2})}, \tag{6}$$

where N_c is the number of data points assigned to the c^{th} component, $\Gamma(\cdot)$ is Gamma function, and

$$r_c = r + N_c, \qquad \nu_c = \nu + N_c,$$

$$\mathbf{u}_c = \frac{r\mathbf{u} + \sum_{n:z_n=c} \mathbf{x}_n}{r + N_c},$$

$$\mathbf{S}_c = \mathbf{S} + \sum_{n:z_n=c} \mathbf{x}_n \mathbf{x}_n^\top + r\mathbf{u}\mathbf{u}^\top - r_c \mathbf{u}_c \mathbf{u}_c^\top, \quad (7)$$

are the posterior Gaussian-Wishart parameters of the c^{th} component. We use a Dirichlet process with concentration parameter η for infinite mixture modeling [5] in the latent space. Then, the probability of $\mathbf{Z}$ is given as follows:

$$p(\mathbf{Z}|\eta) = \frac{\eta^C \prod_{c=1}^{C} (N_c - 1)!}{\eta(\eta+1)\cdots(\eta+N-1)}, \quad (8)$$

where C is the number of components for which $N_c > 0$. The joint distribution is given by

$$\begin{aligned} &p(\mathbf{Y}, \mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}, \boldsymbol{S}, \nu, \mathbf{u}, r, \eta) \\ &= p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) p(\mathbf{X}|\mathbf{Z}, \boldsymbol{S}, \nu, \mathbf{u}, r) p(\mathbf{Z}|\eta), \end{aligned} \quad (9)$$

where factors in the right hand side can be calculated by (1), (6) and (8), respectively.

In summary, the infinite warped mixture model generates observations $\mathbf{Y}$ according to the following generative process:

1. Draw mixture weights $\boldsymbol{\lambda} \sim \text{GEM}(\eta)$
2. For each component $c = 1, \cdots, \infty$
   (a) Draw precision $\mathbf{R}_c \sim \mathcal{W}(\mathbf{S}^{-1}, \nu)$
   (b) Draw mean $\boldsymbol{\mu}_c \sim \mathcal{N}(\mathbf{u}, (r\mathbf{R}_c)^{-1})$
3. For each observed dimension $d = 1, \cdots, D$
   (a) Draw function $f_d(\mathbf{x}) \sim \text{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$
4. For each observation $n = 1, \cdots, N$
   (a) Draw latent assignment $z_n \sim \text{Mult}(\boldsymbol{\lambda})$
   (b) Draw latent coordinates $\mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}_{z_n}, \mathbf{R}_{z_n}^{-1})$
   (c) For each observed dimension $d = 1, \cdots, D$
      i. Draw feature $y_{nd} \sim \mathcal{N}(f_d(\mathbf{x}_n), \beta^{-1})$

Here, GEM(η) is the stick-breaking process [6] that generates mixture weights for a Dirichlet process with parameter η, Mult($\boldsymbol{\lambda}$) represents a multinomial distribution with parameter $\boldsymbol{\lambda}$, $m(\mathbf{x})$ is the mean function of the Gaussian process, and $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^Q$. Figure 2 shows the graphical model representation of the proposed model. Here, we assume a Gaussian for the mixture component, although we could in principle use other distributions such as Student's t-distribution or the Laplace distribution.

The iWMM can be seen as a generalization of either the GPLVM or the infinite Gaussian mixture model

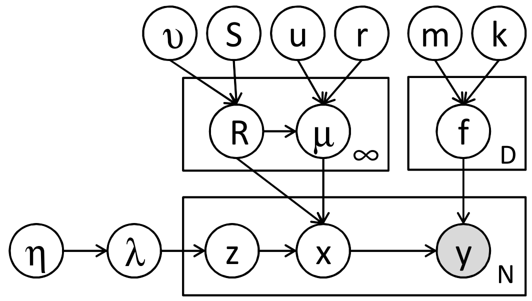


Figure 2: A graphical model representation of the infinite warped mixture model, where the shaded and unshaded nodes indicate observed and latent variables, respectively, and plates indicate repetition.

(iGMM). To be precise, the iWMM with a single fixed spherical Gaussian density on the latent coordinates corresponds to the GPLVM, while the iWMM with fixed direct mapping function $f_d(\mathbf{x}) = x_d$ and $Q = D$ corresponds to the iGMM.

The iWMM offers attractive properties that do not exist in other probabilistic models; principally, the ability to model clusters with nonparametric densities, and to infer a separate dimension for manifold.

4 Inference

We infer the posterior distribution of the latent coordinates $\mathbf{X}$ and cluster assignments $\mathbf{Z}$ using Markov chain Monte Carlo (MCMC). In particular, we alternate collapsed Gibbs sampling of $\mathbf{Z}$, and hybrid Monte Carlo sampling of $\mathbf{X}$. Given $\mathbf{X}$, we can efficiently sample $\mathbf{Z}$ using collapsed Gibbs sampling, integrating out the mixture parameters. Given $\mathbf{Z}$, we can calculate the gradient of the unnormalized posterior distribution of $\mathbf{X}$, integrating over warping functions. This gradient allows us to sample $\mathbf{X}$ using hybrid Monte Carlo.

First, we explain collapsed Gibbs sampling for $\mathbf{Z}$. Given a sample of $\mathbf{X}$, $p(\mathbf{Z}|\mathbf{X}, \mathbf{S}, \nu, \mathbf{u}, r, \eta)$ does not depend on $\mathbf{Y}$. This lets resample cluster assignments, integrating out the iGMM likelihood in close form. Given the current state of all but one latent component z_n, a new value for z_n is sampled from the following probability:

$$p(z_n = c|\mathbf{X}, \mathbf{Z}_{\setminus n}, \boldsymbol{S}, \nu, \mathbf{u}, r, \eta) \propto \begin{cases} N_{c\setminus n} \cdot p(\mathbf{x}_n|\mathbf{X}_{c\setminus n}, \boldsymbol{S}, \nu, \mathbf{u}, r) & \text{existing components} \\ \eta \cdot p(\mathbf{x}_n|\boldsymbol{S}, \nu, \mathbf{u}, r) & \text{a new component} \end{cases} \quad (10)$$

where $\mathbf{X}_c = \{\mathbf{x}_n | z_n = c\}$ is the set of latent coordinates assigned to the c^{th} component, and $\setminus n$ represents the value or set when excluding the n^{th} data point. We can analytically calculate $p(\mathbf{x}_n|\mathbf{X}_{c\backslash n}, \boldsymbol{S}, \nu, \mathbf{u}, r)$ as follows:

$$p(\mathbf{x}_n|\mathbf{X}_{c\backslash n}, \boldsymbol{S}, \nu, \mathbf{u}, r) = \pi^{-\frac{N_{c\backslash n}Q}{2}} \frac{r_{c\backslash n}^{Q/2}|\mathbf{S}_{c\backslash n}|^{\nu_{c\backslash n}/2}}{r_{c\backslash n}'^{Q/2}|\mathbf{S}'_{c\backslash n}|^{\nu'_{c\backslash n}/2}} \prod_{d=1}^{Q} \frac{\Gamma(\frac{\nu'_{c\backslash n}+1-d}{2})}{\Gamma(\frac{\nu_{c\backslash n}+1-d}{2})}, \quad (11)$$

where r'_c, ν'_c, $\mathbf{u}'_c$ and $\mathbf{S}'_c$ represent the posterior Gaussian-Wishart parameters of the c^{th} component when the n^{th} data point is assigned to the c^{th} component. We can efficiently calculate the determinant by using the rank one Cholesky update. In the same way, we can analytically calculate the likelihood for a new component $p(\mathbf{x}_n|\boldsymbol{S}, \nu, \mathbf{u}, r)$.

Hybrid Monte Carlo (HMC) sampling of $\mathbf{X}$ from posterior $p(\mathbf{X}|\mathbf{Z}, \mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{S}, \nu, \mathbf{u}, r)$ requires computing the gradient of the log of the unnormalized posterior $\log p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) + \log p(\mathbf{X}|\mathbf{Z}, \boldsymbol{S}, \nu, \mathbf{u}, r)$. The first term of the gradient can be calculated by

$$\frac{\partial \log p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})}{\partial \mathbf{K}} = -\frac{1}{2}D\mathbf{K}^{-1} + \frac{1}{2}\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T\mathbf{K}^{-1}, \quad (12)$$

and

$$\frac{\partial k(\mathbf{x}_n, \mathbf{x}_m)}{\partial \mathbf{x}_n} = -\frac{\alpha}{\ell^2} \exp\left(-\frac{1}{2\ell^2}(\mathbf{x}_n - \mathbf{x}_m)^\top(\mathbf{x}_n - \mathbf{x}_m)\right)(\mathbf{x}_n - \mathbf{x}_m), \quad (13)$$

using the chain rule. The second term can be calculated as follows:

$$\frac{\partial \log p(\mathbf{X}|\mathbf{Z}, \boldsymbol{S}, \nu, \mathbf{u}, r)}{\partial \mathbf{x}_n} = -\nu_{z_n}\boldsymbol{S}_{z_n}^{-1}(\mathbf{x}_n - \mathbf{u}_{z_n}). \quad (14)$$

We also infer kernel hyperparameters $\boldsymbol{\theta} = \{\alpha, \beta, \ell\}$ via HMC, using the gradient of the log unnormalized posterior with respect to the kernel hyperparameters. The complexity of each iteration of HMC is dominated by the $\mathcal{O}(N^3)$ computation of $\mathbf{K}^{-1}$ [1].

In summary, we obtain samples from the posterior $p(\mathbf{X}, \mathbf{Z}|\mathbf{Y}, \boldsymbol{\theta}, \mathbf{S}, \nu, \mathbf{u}, r, \eta)$ by iterating the following procedures:

1. For each observation $n = 1, \cdots, N$, sample the component assignment z_n by collapsed Gibbs sampling (10).
2. Sample latent coordinates $\mathbf{X}$ and kernel parameters $\boldsymbol{\theta}$ using hybrid Monte Carlo.

[1]This complexity could be improved by making use of an inducing point approximation such as [7, 8]

4.1 Posterior Predictive Density

In the GPLVM, the predictive density of at test point $y^\star$ is usually computed by finding the point $x^\star$ which is most likely to be mapped to $y^\star$, then using the density of $p(x^\star)$ and the Jacobian of the warping at that point to approximately compute the density at $y^\star$. When inference is done by simply optimizing the location of the latent points, this estimation method simply requires solving a single optimization for each $y^\star$.

For our model, we use approximate integration to estimate $p(y^\star)$. This is done for two reasons: First, multiple latent points (possibly from different clusters) can map to the same observed point, meaning the standard method can underestimate $p(y^\star)$. Second, because we do not optimize the latent coordinates but rather sample them, we would need to perform optimizations for each $p(y^\star)$ separately for each sample. Our method gives estimates for all $p(\mathbf{y}^\star)$ at once, but may not be accurate in very high dimensions.

The posterior density in the observed space given the training data is simply:

$$\begin{aligned} p(\mathbf{y}^\star|\mathbf{Y}) &= \iint p(\mathbf{y}^\star, \mathbf{x}^\star, \mathbf{X}|\mathbf{Y})d\mathbf{x}^\star d\mathbf{X} \\ &= \iint p(\mathbf{y}^\star|\mathbf{x}^\star, \mathbf{X}, \mathbf{Y})p(\mathbf{x}^\star|\mathbf{X}, \mathbf{Y})p(\mathbf{X}|\mathbf{Y})d\mathbf{x}^\star d\mathbf{X}. \end{aligned} \quad (15)$$

We approximate $p(\mathbf{X}|\mathbf{Y})$ using the samples from the Gibbs and hybrid Monte Carlo samplers. We approximate $p(\mathbf{x}^\star|\mathbf{X}, \mathbf{Y})$ by sampling points from the latent mixture and warping them, using the following procedure:

1. Draw latent assignment
$z^\star \sim \text{Mult}(\frac{N_1}{N+\eta}, \cdots, \frac{N_C}{N+\eta}, \frac{\eta}{N+\eta})$
2. Draw precision matrix
$\mathbf{R}^\star \sim \mathcal{W}(\mathbf{S}_{z^\star}^{-1}, \nu_{z^\star})$
3. Draw mean
$\boldsymbol{\mu}^\star \sim \mathcal{N}(\mathbf{u}_{z^\star}, (r_{z^\star}\mathbf{R}^\star)^{-1})$
4. Draw latent coordinates
$\mathbf{x}^\star \sim \mathcal{N}(\boldsymbol{\mu}^\star, \mathbf{R}^{\star -1})$

When a new component $C+1$ is assigned to $z^\star$, the prior Gaussian-Wishart distribution is used for sampling in steps 2 and 3. The first factor of (15) can be calculated by

$$p(\mathbf{y}^\star|\mathbf{x}^\star, \mathbf{X}, \mathbf{Y}) = \mathcal{N}(\mathbf{k}^{\star\top}\mathbf{K}^{-1}\mathbf{Y}, k(\mathbf{x}^\star, \mathbf{x}^\star) - \mathbf{k}^{\star\top}\mathbf{K}^{-1}\mathbf{k}^\star), \quad (16)$$

where $\mathbf{k}^\star = (k(\mathbf{x}^\star, \mathbf{x}_1), \cdots, k(\mathbf{x}^\star, \mathbf{x}_N))^\top$. Each step of this procedure is exact. Since the observations $\mathbf{y}^\star$ are conditionally normally distributed, each one adds a smooth local contribution to the empirical Monte Carlo estimate of the posterior density, as opposed to a point mass. This procedure was used to generate the plots of posterior density in Figures 1, 4, and 6.

5 Related work

Latent Variable Models The GPLVM is effective as a nonlinear latent variable model in a wide variety of applications [3, 9, 10]. The latent positions $\mathbf{X}$ in the GPLVM are typically obtained by maximum a posteriori estimation or variational Bayesian inference [11], placing a single fixed spherical Gaussian prior on $\mathbf{x}$. A prior which penalizes a high-dimensional latent space is introduced by [12], in which the latent variables and their intrinsic dimensionality are simultaneously optimized. The iWMM can also infer the intrinsic dimensionality of each manifolds: inferring the Gaussian covariance for each latent cluster allows the variance along irrelevant dimensions to become small. Because each latent cluster has a different set of parameters, the effective dimension of each cluster can vary, allowing manifolds of different dimension in the observed space. This ability is demonstrated in Figure 4 (c).

The iWMM can also be viewed as a generalization of the mixture of probabilistic principle component analyzers [13], or mixture of factor analyzers [14], where the linear mapping of the mixtures is generalized to a nonlinear mapping by Gaussian processes, and number of components is infinite.

Clustering Methods There exist non-probabilistic clustering methods which can find clusters with complex shapes, such as spectral clustering [15] and nonlinear manifold clustering [16, 17]. Spectral clustering finds clusters by first forming a similarity graph, then finding a low-dimensional latent representation using the graph, and finally, clustering the latent coordinates via k-means. The performance of spectral clustering depends on parameters which are usually set manually, such as the number of clusters, the number of neighbors, and the variance parameter used for constructing the similarity graph. In contrast, the iWMM infers such parameters automatically. One of the main advantages of the iWMM over these methods is that there is no need to construct a similarity graph.

The kernel Gaussian mixture model [18] can also find non-Gaussian shaped clusters. This model estimates a GMM in the implicit high-dimensional feature space defined by the kernel mapping of the observed space. However, the kernel GMM uses a fixed nonlinear mapping function, with no guarantee that the latent points will be well-modeled by a GMM. In contrast, the iWMM infers the mapping function such that the latent co-ordinates will be well-modeled by a GMM.

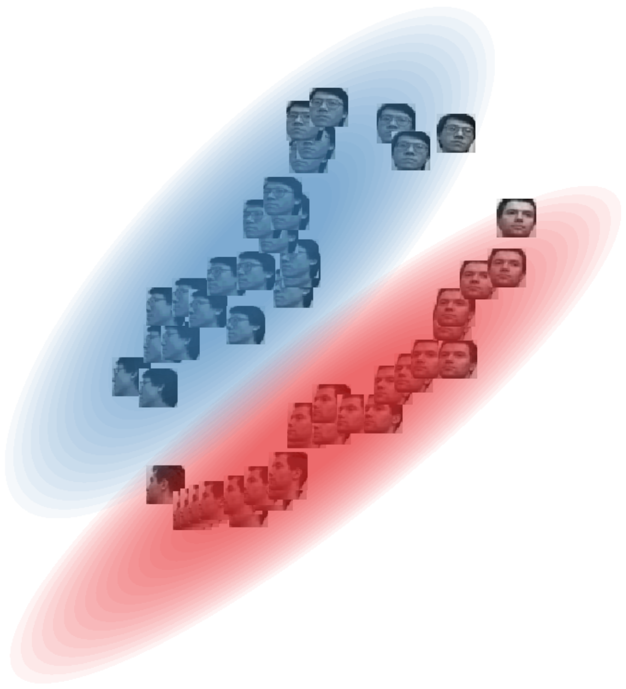


Figure 3: A sample from the 2-dimensional latent space when modeling a series of 32x32 face images. Our model correctly discovers that the data consists of two separate manifolds, both approximately one-dimensional, which share the same head-turning structure.

For one-dimensional data, [19] introduce a nonparametric model of *unimodal* clusters, where each cluster's density function decreases away from its mode.

6 Experimental results

Clustering Faces We first examined our model's ability to model images without pre-processing. We constructed a dataset consisting of 50 greyscale 32x32 pixel images of two individuals from the UMIST faces dataset [20]. Both series of images capture a person turning his head to the right. Figure 3 shows a sample from the posterior over the latent coordinates and density model. The model has recovered three relevant, interpretable features of the dataset. First, that there are two distinct faces. Second, that each set of images lies approximately along a smooth one-dimensional manifold. Third, that the two manifolds share roughly the same structure: the front-facing images of both individuals lie close to one another, as do

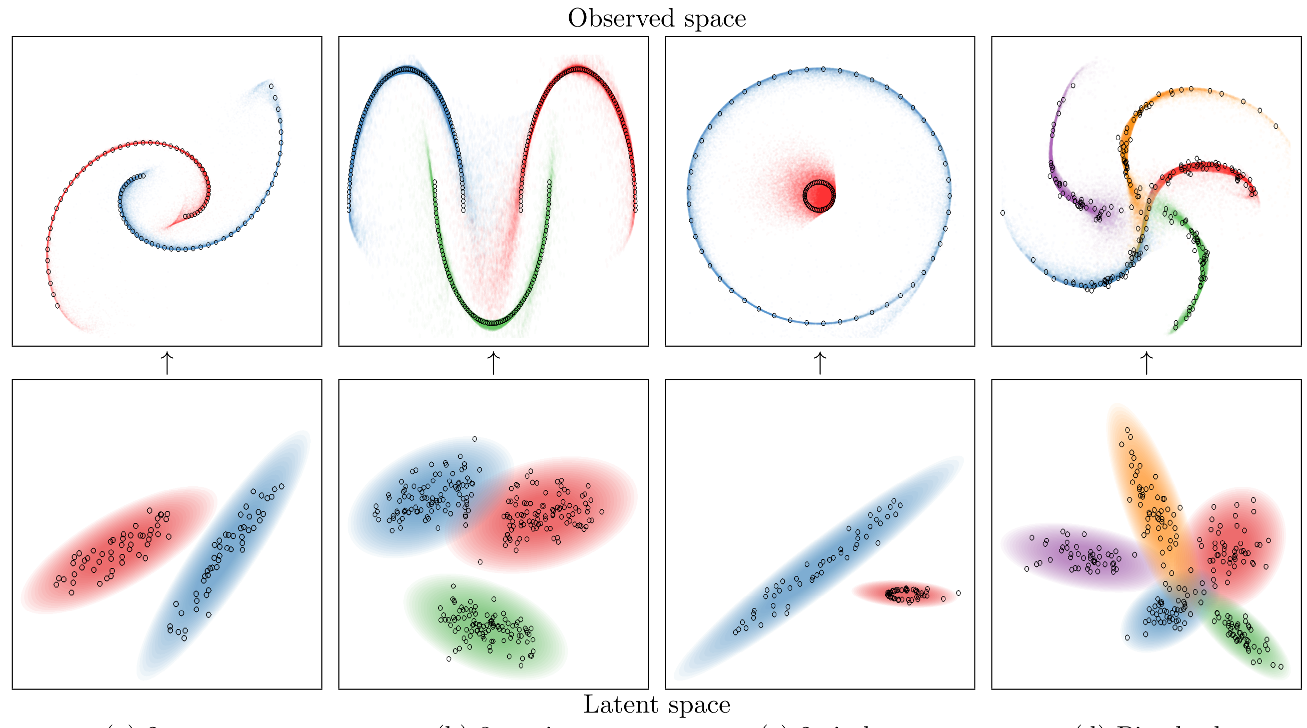


Figure 4: Top row: The observed, unlabeled data points, and the clusters inferred by the iWMM. Bottom row: Latent coordinates and Gaussian components, shown for a single sample from the posterior. Each point in the latent space corresponds to a point in the observed space. This figure is best viewed in color.

the side-facing images.

6.1 Synthetic Datasets

Next, we demonstrate the proposed model on the four synthetic datasets shown in Figure 4. None of these four datasets can be appropriately clustered by Gaussian mixture models (GMM). For example, consider the 2-curve data shown in Figure 4 (a), where 100 data points lie in one of two curved lines in a two-dimensional observed space. A GMM with two components cannot separate the two curved lines, while a GMM with many components could separate the two lines only by breaking each line into many clusters. In contrast, in the iWMM, the two non-Gaussian-shaped clusters in the observed space were represented by two Gaussian-shaped clusters in the latent space, as shown at the bottom row of Figure 4 (a). The iWMM separated the two curved lines by nonlinearly warping two Gaussians from the latent space to the observed space.

Figure 4 (c) shows an interesting manifold learning challenge: a dataset consisting of two circles. The outer circle is modeled in the latent space by a Gaussian with effectively one degree of freedom. This linear topology fits the outer circle in the observed space by bending the two ends until they overlap. In contrast, the sampler fails to discover the 1D topology of the inner circle, modeling it with a 2D manifold instead. This example demonstrates that each cluster manifold in the iWMM can have a different effective dimension.

6.2 Mixing

An interesting side-effect of learning the number of latent clusters is that this added flexibility can help the sampler escape local minima, helping the sampler to mix properly. Figure 5 shows the samples of the latent coordinates and clusters of the iWMM over time, when modeling the 2-curve data. 5(a) shows the latent coordinates initialized at the observed coordinates, starting with one latent component. At the 500th iteration 5(b), each curved line is modeled by two components. At the 1800th iteration 5(c), the left curved line is modeled by a single component. At the 3000th iteration 5(d), the right curved line is also modeled by a single component, and the dataset is appropriately clustered. This configuration was relatively stable, and a similar state was found at the 5000th iteration.

6.3 Density Estimation

Figure 6 (a) shows the posterior density in the observed space inferred by the iWMM on the 2-curve

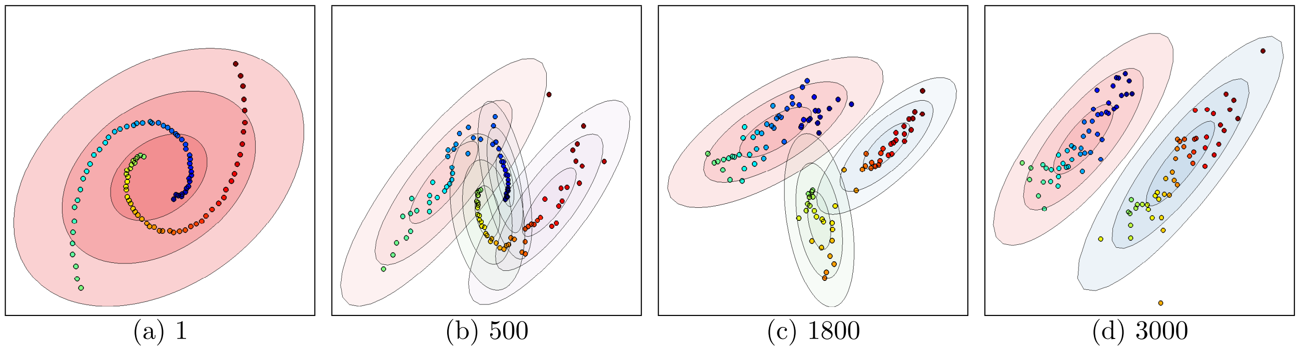


Figure 5: The inferred infinite GMMs over iterations in the two-dimensional latent space with the iWMM using the 2-curve data. Labels indicate the number of iterations of the sampler, and the color of each point represents its ordering in the observed coordinates.

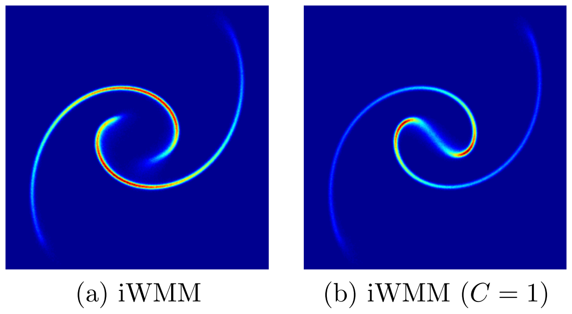


Figure 6: The posterior density in the observed space with the 2-curve data inferred by the iWMM (a), and that inferred by the iWMM with one component (b).

data, computed using 1000 samples from the Markov chain. The two separate manifolds of high density implied by the two curved lines was recovered by the iWMM. Note also that the density along the manifold varies with the density of data shown in Figure 4 (a). This result can be compared to a special case of our model, which uses only a single Gaussian to model the latent coordinates instead of an infinite GMM. Figure 6 (b) shows that the result of the iWMM with $C = 1$, where posterior is forced to place significant density connecting the two clusters. Figure 6 (b) shows that the single-cluster variant of the iWMM posterior is forced to place significant density connecting the two clusters.

6.4 Visualization

Next, we briefly investigate the potential of the iWMM for visualization. Figure 7 (a) shows the latent coordinates obtained by averaging over 1000 samples from the posterior of the iWMM. Because rotating the latent coordinates does not change their probability, averaging may not be an adequate way to summarize the posterior. However, we show this result in order to show the characteristics of latent coordinates obtained by the iWMM. The estimated latent coordinates are clearly separated, and they form two straight lines. This result indicates that in some cases, the iWMM can recover the topology of the data before it has been warped into a manifold. For comparison, Figure 7 (b) shows the latent coordinates estimated by the iWMM when forced to use a single cluster: the latent coordinates lie in two sections of a single straight line. Figures 7 (c) and (d) show the latent coordinates estimated by the GPLVM when optimizing or integrating out the latent coordinates, respectively. Recall that the iWMM ($C = 1$) is a more flexible model than the GPLVM, since the GPLVM enforces a spherical covariance in the latent space. These methods did not unfold the two curved lines, since the effective dimension of their latent representation is fixed beforehand. In contrast, the iWMM effectively formed a low-dimensional representation in the latent space.

Regardless of the dimension of the latent space, the iWMM will tend to model each cluster with as low-dimensional a Gaussian as possible. This is because, if the data in a cluster can be made to lie in a low-dimensional plane, a narrowly-shaped Gaussian will assign the latent coordinates much higher likelihood than a spherical Gaussian.

6.5 Clustering Performance

We more formally evaluated the density estimation and clustering performance of the proposed model using four real datasets: iris, glass, wine and vowel, obtained from LIBSVM multi-class datasets [21], in addition to the four synthetic datasets shown above: 2-curve, 3-semi, 2-circle and Pinwheel [22]. The statistics of these datasets are summarized in Table 1. In each experiment, we show the results of 20-fold cross-validation. Results in bold are not significantly different from the best performing method in each column

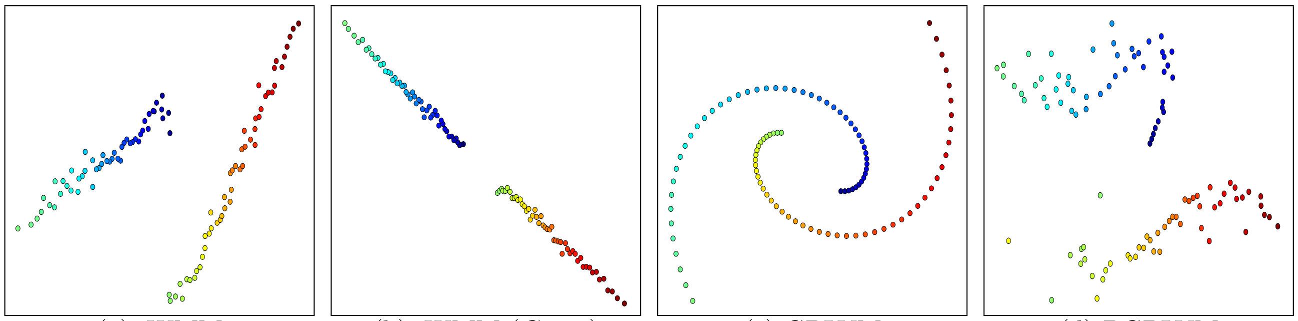

(a) iWMM (b) iWMM ($C = 1$) (c) GPLVM (d) BGPLVM

Figure 7: The estimated latent coordinates of the 2-curve data by (a) iWMM, (b) iWMM ($C = 1$), (c) GPLVM, and (d) Bayesian GPLVM.

Table 1: The statistics of datasets used for evaluation.

	2-curve	3-semi	2-circle	Pinwheel	Iris	Glass	Wine	Vowel
number of samples: N	100	300	100	250	150	214	178	528
observed dimensionality: D	2	2	2	2	4	9	13	10
number of clusters: C	2	3	2	5	3	7	3	11

Table 2: Average Rand index for evaluating clustering performance.

	2-curve	3-semi	2-circle	Pinwheel	Iris	Glass	Wine	Vowel
iGMM	0.52	0.79	0.83	0.81	0.78	0.60	0.72	**0.76**
iWMM(Q=2)	**0.86**	**0.99**	**0.89**	**0.94**	**0.81**	**0.65**	0.65	0.50
iWMM(Q=D)	**0.86**	**0.99**	**0.89**	**0.94**	0.77	0.62	**0.77**	**0.76**

according to a paired t-test.

Table 2 compares the clustering performance of the iWMM with the iGMM, quantified by the Rand index [23], which measures the correspondence between inferred clusters and true clusters. The iGMM is another probabilistic generative model commonly used for clustering, which can be seen as a special case of the iWMM in which the Gaussian clusters are not warped. These experiments demonstrate the extent to which nonparametric cluster shapes allow a mixture model to recover more meaningful clusters.

Table 3 lists average test log likelihood, comparing the proposed models with kernel density estimation (KDE), and the infinite Gaussian mixture model (iGMM). In KDE, the kernel width is estimated by maximizing the leave-one-out log densities. Since the manifold on which the observed data lies can be at most D-dimensional, we set the latent dimension Q equal to the observed dimension D in iWMMs. We also include the $Q = 2$ case in an attempt to characterize how much modeling power is lost by forcing the latent representation to be visualizable. The proposed models achieved high test log likelihoods compared with the KDE and iGMM.

6.6 Source code

Code to reproduce all the above experiments is available at `github.com/duvenaud/warped-mixtures`.

7 Future work

The Dirichlet process mixture of Gaussians in the latent space of our model could easily be replaced by a more sophisticated density model, such as a hierarchical Dirichlet process [24], or a Dirichlet diffusion tree [25]. Another straightforward extension of our model would be making inference more scalable by using sparse Gaussian processes [7, 8] or more advanced hybrid Monte Carlo methods [26]. An interesting but more complex extension of the iWMM would be a semi-supervised version of the model. The iWMM could allow label propagation along regions of high density in the latent space, even if those regions were stretched along low-dimensional manifolds in the observed space. Another natural extension would be to allow a separate warping for each cluster, which would also improve inference speed.

Table 3: Average test log likelihood for evaluating density estimation performance.

	2-curve	3-semi	2-circle	Pinwheel	Iris	Glass	Wine	Vowel
KDE	-2.47	-0.38	-1.92	-1.47	$\mathbf{-1.87}$	1.26	-2.73	$\mathbf{6.06}$
iGMM	-3.28	-2.26	-2.21	-2.12	-1.91	3.00	$\mathbf{-1.87}$	-0.67
iWMM(Q=2)	$\mathbf{-0.90}$	$\mathbf{-0.18}$	$\mathbf{-1.02}$	$\mathbf{-0.79}$	$\mathbf{-1.88}$	$\mathbf{5.76}$	$\mathbf{-1.96}$	$\mathbf{5.91}$
iWMM(Q=D)	$\mathbf{-0.90}$	$\mathbf{-0.18}$	$\mathbf{-1.02}$	$\mathbf{-0.79}$	$\mathbf{-1.71}$	$\mathbf{5.70}$	-3.14	-0.35

8 Conclusion

In this paper, we introduced a simple generative model of non-Gaussian density manifolds which can infer nonlinearly separable clusters, low-dimensional representations of varying dimension per cluster, and density estimates which smoothly follow data contours. We then introduced an efficient sampler for this model which integrates out both the cluster parameters and the warping function exactly. We further demonstrated that allowing non-parametric cluster shapes improves clustering performance over the Dirichlet process Mixture of Gaussians.

Many methods have been proposed which can perform some combination of clustering, manifold learning, density estimation and visualization. We demonstrated that a simple but flexible probabilistic generative model can perform well at all these tasks.

Acknowledgements

The authors would like to thank Dominique Perrault-Joncas, Carl Edward Rasmussen, and Ryan Prescott Adams for helpful discussions.

References

[1] C.E. Rasmussen. The infinite Gaussian mixture model. *Advances in Neural Information Processing Systems*, 12(5.2):2, 2000.

[2] C.E. Rasmussen and CKI Williams. Gaussian Processes for Machine Learning. *The MIT Press, Cambridge, MA, USA*, 2006.

[3] N.D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. *Advances in Neural Information Processing Systems*, 16:329–336, 2004.

[4] H. Nickisch and C. Rasmussen. Gaussian mixture modeling with Gaussian process latent variable models. *Pattern Recognition*, pages 272–282, 2010.

[5] S.N. MacEachern and P. Müller. Estimating mixture of Dirichlet process models. *Journal of Computational and Graphical Statistics*, pages 223–238, 1998.

[6] Jayaram Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[7] J. Quiñonero-Candela and C.E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959, 2005.

[8] E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. *Advances in Neural Information Processing Systems*, 2006.

[9] M. Salzmann, R. Urtasun, and P. Fua. Local deformation models for monocular 3D shape recovery. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, pages 1–8, 2008.

[10] N.D. Lawrence and R. Urtasun. Non-linear matrix factorization with Gaussian processes. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 601–608. ACM, 2009.

[11] M. Titsias and N. Lawrence. Bayesian Gaussian process latent variable model. *AISTATS*, 2010.

[12] A. Geiger, R. Urtasun, and T. Darrell. Rank priors for continuous non-linear dimensionality reduction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 880–887. IEEE, 2009.

[13] M.E. Tipping and C.M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural computation*, 11(2):443–482, 1999.

[14] Z. Ghahramani and M.J. Beal. Variational inference for Bayesian mixtures of factor analysers. *Advances in Neural Information Processing Systems*, 12:449–455, 2000.

[15] A.Y. Ng, M.I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 2:849–856, 2002.

[16] W. Cao and R. Haralick. Nonlinear manifold clustering by dimensionality. In *International Conference on Pattern Recognition (ICPR)*, volume 1, pages 920–924. IEEE, 2006.

[17] Ehsan Elhamifar and René Vidal. Sparse manifold clustering and embedding. In *Advances*

in Neural Information Processing Systems, pages 55–63, 2011.

[18] J. Wang, J. Lee, and C. Zhang. Kernel trick embedded Gaussian mixture model. In *Algorithmic Learning Theory*, pages 159–174. Springer, 2003.

[19] Carlos E Rodríguez and Stephen G Walker. Univariate bayesian nonparametric mixture modeling with unimodal kernels. *Statistics and Computing*, pages 1–15, 2012.

[20] Daniel B Graham and Nigel M Allinson. Characterizing virtual eigensignatures for general purpose face recognition. *Face Recognition: From Theory to Applications*, 163:446–456, 1998.

[21] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, 2011.

[22] R.P. Adams and Z. Ghahramani. Archipelago: nonparametric Bayesian semi-supervised learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009.

[23] W.M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, pages 846–850, 1971.

[24] Y.W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[25] R.M. Neal. Density modeling and clustering using dirichlet diffusion trees. *Bayesian Statistics*, 7:619–629, 2003.

[26] Y. Zhang and C. Sutton. Quasi-Newton Markov chain Monte Carlo. *Advances in Neural Information Processing Systems*, pages 2393–2401, 2011.

The Lovász-Bregman Divergence and connections to rank aggregation, clustering, and web ranking

Rishabh Iyer
Dept. of Electrical Engineering
University of Washington
Seattle, WA-98175, USA

Jeff Bilmes
Dept. of Electrical Engineering
University of Washington
Seattle, WA-98175, USA

Abstract

We extend the recently introduced theory of Lovász Bregman (LB) divergences [19] in several ways. We show that they represent a distortion between a "score" and an "ordering", thus providing a new view of rank aggregation and order based clustering with interesting connections to web ranking. We show how the LB divergences have a number of properties akin to many permutation based metrics, and in fact have as special cases forms very similar to the Kendall-τ metric. We also show how the LB divergences subsume a number of commonly used ranking measures in information retrieval, like NDCG [22] and AUC [35]. Unlike the traditional permutation based metrics, however, the LB divergence naturally captures a notion of "confidence" in the orderings, thus providing a new representation to applications involving aggregating scores as opposed to just orderings. We show how a number of recently used web ranking models are forms of Lovász Bregman rank aggregation and also observe that a natural form of Mallow's model using the LB divergence has been used as conditional ranking models for the "Learning to Rank" problem.

1 Introduction

The Bregman divergence first appeared in the context of relaxation techniques in convex programming [5], and has found numerous applications as a general framework in clustering [3], proximal minimization ([7]), and others. Many of these applications are due to the nice properties of the Bregman divergence, and the fact that they are parameterized by a single convex function. They also generalize a large class of divergences between vectors.

In this paper, we investigate a specific class of Bregman divergences, parameterized via the Lovász extension of a submodular function. Submodular functions are a special class of discrete functions with interesting properties. Let V refer to a finite ground set $\{1, 2, \dots, |V|\}$. A set function $f : 2^V \to \mathbb{R}$ is submodular if $\forall S, T \subseteq V$, $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$. Submodular functions have attractive properties that make their exact or approximate optimization efficient and often practical [17, 21]. They also naturally arise in many problems in machine learning, computer vision, economics, operations research, etc. A link between convexity and submodularity is seen via the Lovász extension ([13, 29]) of the submodular function. While submodular functions are growing phenomenon in machine learning, recently there has been an increasing set of applications for the Lovász extension. In particular, recent work [1, 2] has shown nice connections between the Lovász extension and structured sparsity inducing norms.

This work is concerned with yet another application of the Lovász extension, in the form of the Lovász-Bregman divergence. This was first introduced in Iyer & Bilmes [19], in the context of clustering ranked vectors. We extend our work in several ways, mainly theoretically, by both showing a number of connections to the permutation based metrics, to rank aggregation, to rank based clustering and to the "Learning to Rank" problem in web ranking.

1.1 Motivation

The problems of rank aggregation and rank based clustering are ubiquitous in machine learning, information retrieval, and social choice theory. Below is a partial list of some of these applications.

Meta Web Search: We are given a collection of search engines, each providing a ranking or a score vector, and the task is to aggregate these to generate a combined result [27].

Learning to Rank: The "Learning to rank" problem, which is a fundamental problem in machine learning, involves constructing a ranking model from training data. This problem has gained significant interest in web ranking and information retrieval [28].

Voter or Rank Clustering: This is an important problem in social choice theory, where each voter provides a ranking or assigns a score to every item. A natural problem here is to meaningfully combine these rankings [26]. Sometimes however the population is heterogeneous and a mixture of distinct populations, each with its own aggregate representative, fits better.

Combining Classifiers and Boosting: There has been an increased interest in combining the output of different systems in an effort to improve performance of pattern classifiers, something often used in Machine Translation [34] and Speech Recognition[24]. One way of doing this [27] is to treat the output of every classifier as a ranking and combine the individual rankings of weak classifiers to obtain the overall classification. This is akin to standard boosting techniques [16], except that we consider rankings rather than just the valuations.

1.2 Permutation Based Distance Metrics

First a bit on notation – a permutation σ is a bijection from $[n] = \{1, 2, \cdots, n\}$ to itself. Given a permutation σ, we denote σ^{-1} as the inverse permutation such that $\sigma(i)$ is the item assigned rank i, while $\sigma^{-1}(j)$ is the rank[1] assigned to item j and hence $\sigma(\sigma^{-1}(i)) = i$. We shall use σ_x to denote a permutation induced through the ordering of a vector x such that $x(\sigma_x(1)) \geq x(\sigma_x(2)) \cdots \geq x(\sigma_x(n))$. Without loss of generality, we assume that the permutation is defined via a decreasing order of elements. We shall use $v(i), v[i]$ and v_i interchangeably to denote the i-th element in v. Given two permutations σ, π we can define $\sigma\pi$ as the combined permutation, such that $\sigma\pi(i) = \sigma(\pi(i))$. Also given a vector x and a permutation σ, define $x\sigma$ such that $x\sigma(i) = x(\sigma(i))$. We also define σx as $\sigma x(i) = x(\sigma^{-1}(i))$.

Recently a number of papers [27, 26, 23, 31] have addressed the problem of combining rankings using permutation based distance metrics. Denote $\mathbf{\Sigma}$ as the set of permutations over $[n]$. Then $d : \mathbf{\Sigma} \times \mathbf{\Sigma} \to \mathbb{R}_+$ is a permutation based distance metric if it satisfies the usual notions of a metric, viz. , $\forall \sigma, \pi, \tau \in \mathbf{\Sigma}, d(\sigma, \pi) \geq 0$ and $d(\sigma, \pi) = 0$ iff $\sigma = \pi$, $d(\sigma, \pi) = d(\pi, \sigma)$ and $d(\sigma, \pi) \leq d(\sigma, \tau) + d(\tau, \pi)$. In addition, to represent a distance amongst permutations, another property which is usually required is that of left invariance to reorderings, i.e., $d(\sigma, \pi) = d(\tau\sigma, \tau\pi)$[2]. The most standard notion of a permutation based distance metric is the Kendall τ metric [23]:

$$d_T(\sigma, \pi) = \sum_{i,j,i<j} I(\sigma^{-1}\pi(i) > \sigma^{-1}\pi(j)) \qquad (1)$$

Where $I(.)$ is the indicator function. This distance metric represents the number of swap operations required to convert a permutation σ to π. It's not hard to see that it is a metric and it satisfies the ordering invariance property. Other often used metrics include the Spearman's footrule d_S and rank correlation d_R [10]:

$$d_S(\sigma, \pi) = \sum_{i=1}^{n} |\sigma^{-1}(i) - \pi^{-1}(i)| \qquad (2)$$

$$d_R(\sigma, \pi) = \sum_{i=1}^{n} (\sigma^{-1}(i) - \pi^{-1}(i))^2 \qquad (3)$$

A natural extension to a ranking model is the Mallows model [30], which is an exponential model defined based on these permutation based distance metrics. This is defined as:

$$p(\pi|\theta, \sigma) = \frac{1}{Z(\theta)} \exp(-\theta d(\pi, \sigma)), \quad \text{with} \quad \theta \geq 0. \qquad (4)$$

This model has been generalized by [14] and also extended to multistage ranking by [15]. Lebanon and Lafferty [27] were amongst the first to use these models in machine learning by proposing an extended mallows model [14] to combine rankings in a manner like adaboost [16]. Similarly Meila *et al* [31] use the generalized Mallows model to infer the optimal combined ranking. Another related though different problem is clustering ranked data, investigated by [32], where they provide a k-means style algorithm. This was also extended to a machine learning context by [6].

1.3 Score based Permutation divergences

In this paper, we motivate another class of divergences, which capture the notion of distance between permutations. Unlike the permutation based distance metrics, however, these are distortion functions between a "score" and a permutation. This, as we shall see, offers a new view of rank aggregation and order based clustering problems. We shall also see a number of interesting connections to web ranking.

Consider a scenario where we are given a collection of scores $x^1, x^2, \cdots, x^n$ as opposed to just a collection of orderings – i.e., each x^i is an ordered vector and not just a permutation. This occurs in a number of real world applications. For example, in the application of combining classifiers [27], the classifiers often output scores (in the form of say normalized confidence or probability distributions). While the rankings themselves are informative, it is often more beneficial to use the additional information in the form of scores

[1]This is opposite from the convention used in [27, 26, 23, 31] but follows the convention of [17].

[2]While in the literature this is called right invariance, we have left invariance due to our notation

if available. This in some sense combines the approach of Adaboost [16] and Cranking [27], since the former is concerned only with the scores while the latter takes only the orderings. The case of voting is similar, where each voter might assign scores to every candidate (which can sometimes be easier than assigning an ordering). This also applies to web-search where often the individual search engines (or possibly features) provide a confidence score for each webpage. Since these applications provide both the valuations and the rankings, we call these *score based ranking applications.*

A *score based permutation divergence* is defined as follows. Given a convex set $\mathbb{S}$, denote $d : \mathbb{S} \times \mathbf{\Sigma} \to \mathbb{R}_+$ as a score based permutation divergence if $\forall x \in \mathbb{S}, \sigma \in \mathbf{\Sigma}, d(x||\sigma) \geq 0$ and $d(x||\sigma) = 0$ if and only if $\sigma_x = \sigma$. Another desirable property is that of left invariance, viz. $d(x||\sigma) = d(\tau x||\tau\sigma), \forall \tau, \sigma \in \mathbf{\Sigma}, x \in \mathbb{S}$.

It is then immediately clear how the score based permutation divergence naturally models the above scenario. The problem becomes one of finding a representative ordering, i.e., find a permutation σ that minimizes the average distortion to the set of points $x^1, \cdots, x^n$. Similarly, in a clustering application, to cluster a set of ordered scores, a score based permutation divergence fits more naturally. The representatives for each cluster are permutations, while the objects themselves are ordered vectors. Notice that in both cases, a purely permutation based distance metric would completely ignore the values, and just consider the induced orderings or permutations. To our knowledge, this work is the first time that the notion of a score based permutation divergence has been introduced formally, thus providing a novel view to the rank aggregation and rank based clustering problems.

1.4 Our Contributions

In this paper, we investigate several theoretical properties of one such score based permutation divergence – the LB divergence. This work builds on our previous work [19], where we introduce the Lovász-Bregman divergence. Our focus therein is mainly on the connections between the Lovász Bregman and a discrete Bregman divergence connected with submodular functions and we also provide a k-means framework for clustering ordered vectors. In the present paper, however, we make the connection to rank aggregation and clustering more precise, by motivating the class of score based permutation divergences and showing relations to permutation based metrics and web ranking.

The following are some of our main results. We introduce a novel notion of the generalized Bregman divergence based on a "subgradient map". While this is of independent theoretical interest, it helps us characterize the Lovász-Bregman divergence. We then show that the LB divergence is indeed a score based permutation divergence with several similarities to permutation based metrics. In fact, we show that a form of weighted Kendall τ, and a form related to the Spearman's Footrule, occurs as instances of the Lovász-Bregman divergences. We also show how a number of loss functions used in IR and web ranking like the Normalized Discounted Cumulative Gain (NDCG) [22] and the Area Under the Curve (AUC) [35] occur as instances of the LB. We then demonstrate some unique properties of the Lovász Bregman divergence not present in permutation-distance metrics. Notable amongst these are the properties that the Lovász-Bregman naturally captures a notion of "confidence" of an ordering, and exhibits a priority for higher rankings, both of which are desirable in score based ranking applications. We then define a Lovász-Mallows model as a conditional model over both the scores and the ranking. We finally connect the Lovász Bregman to rank aggregation and rank based clustering. We show in fact that a number of ranking models for web ranking used in the past are instances of Lovász Bregman rank aggregation. We moreover show that a number of conditional models used in the past for learning to Rank are closely related to the Lovász-Mallows model.

2 The Lovász Bregman divergences

In this section, we shall briefly review the Lovász extension and define forms of the generalized Bregman and the LB divergence. We only state the main results here and for a more extensive discussion, refer to [20].

2.1 The Generalized Bregman divergences

The notion used in this section follows from [33, 37]. We denote ϕ as a proper convex function (i.e., it's domain is non-empty and it does not take the value $-\infty$), reint(.) and dom(.) as the relative interior and domain respectively. A subgradient g at $y \in \text{dom}(\phi)$ is such that for any $x, \phi(x) \geq \phi(y) + \langle g, x - y\rangle$ and the set of all subgradients at y is the subdifferential and is denoted by $\partial_\phi(y)$.

The Taylor series approximation of a twice differentiable convex function provides a natural way of generating a Bregman divergence ([5]). Given a twice differentiable and strictly convex function ϕ:

$$d_\phi(x, y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y\rangle. \qquad (5)$$

In order to extend this notion to non-differentiable convex functions, generalized Bregman divergences have been proposed [37, 25]. While gradients no longer exist at points of non-differentiability, the directional derivatives exist in the relative interior of the domain of ϕ, as long as the function is finite. Hence a natural formulation is to replace the gradient by the directional derivative, a notion which has been pursued in [37, 25].

In this paper, we view the generalized Bregman divergences slightly differently, in a way related to the approach in [18]. In order to ensure that the subgradients exist, we only consider the relative interior of the domain. Then define $\mathcal{H}_\phi(y)$ as a subgradient-map such that $\forall y \in \text{reint}(\text{dom}(\phi)), \mathcal{H}_\phi(y) \in \partial_\phi(y)$. Then given $x \in \text{dom}(\phi), y \in \text{reint}(\text{dom}(\phi))$ and a subgradient map $\mathcal{H}_\phi$, we define the generalized Bregman divergence as:

$$d_\phi^{\mathcal{H}_\phi}(x, y) = \phi(x) - \phi(y) - \langle \mathcal{H}_\phi(y), x - y\rangle \quad (6)$$

When ϕ is differentiable, notice that $\partial_\phi(y) = \{\nabla\phi(y)\}$ and hence $\mathcal{H}_\phi(y) = \nabla(y)$.

2.2 Properties of the Lovász Extension

We review some important theoretical properties of the Lovász extension. Given any vector $y \in [0,1]^n$ and it's associated permutation σ_y, define $S_j^{\sigma_y} = \{\sigma_y(1), \cdots, \sigma_y(j)\}$ for $j \in [n]$. Notice that in general σ_y need not be unique (it will be unique only if y is totally ordered), and hence let $\mathbf{\Sigma}_y$ represent the set of all possible permutations with this ordering. Then the Lovász extension of f is defined as:

$$\hat{f}(y) = \sum_{j=1}^{n} y[\sigma_y(j)](f(S_j^{\sigma_y}) - f(S_{j-1}^{\sigma_y})) \quad (7)$$

This is also called the Choquet integral [9] of f. Though σ_y might not be unique, the Lovász extension is actually unique. Furthermore, $\hat{f}$ is convex if and only if f is submodular. In addition, the Lovász extension is also tight on the vertices of the hypercube, in that $f(X) = \hat{f}(1_X), \forall X \subseteq V$ (where 1_X is the characteristic vector of X, i.e., $1_X(j) = I(j \in X)$) and hence it is a valid continuous extension. The Lovász extension is in general a non-smooth convex function, and hence there does not exist a unique subgradient at every point. The following result due [17, 13] provides a characterization of the extreme points of the Lovász subdifferential polyhedron $\partial\hat{f}(y)$:

Lemma 2.1. *[17, 13] For a submodular function f, a vector y and a permutation $\sigma_y \in \mathbf{\Sigma}_y$, a vector $h_{\sigma_y}^f$ defined as:*

$$h_{\sigma_y}^f(\sigma_y(j)) = f(S_j^{\sigma_y}) - f(S_{j-1}^{\sigma_y}), \forall j \in \{1, 2, \cdots, n\}$$

forms an extreme point of $\partial\hat{f}(y)$. Also, the number of extreme points of $\partial\hat{f}(y)$ is $|\mathbf{\Sigma}_y|$.

Notice that the extreme subgradients are parameterized by the permutation σ_y and hence we refer to them as $h_{\sigma_y}^f$. Seen in this way, the Lovász extension then takes an extremely simple form: $\hat{f}(w) = \langle h_{\sigma_w}^f, w\rangle$.

We now point out an interesting property related to the extreme subgradients of $\hat{f}$. Define $\mathcal{P}(\sigma)$ as a $n-$simplex corresponding to a permutation σ (or chain $\mathcal{C}^\sigma : \emptyset \subset S_1^\sigma \subset \cdots \subset S_n^\sigma = V$). In other words, $\mathcal{P}(\sigma) = \text{conv}(1_{S_i^\sigma}, i = 1, 2, \cdots, n)$. It's easy to see that $\mathcal{P}(\sigma) \subseteq [0,1]^n$.

Lemma 2.2. *(Lemma 6.19, [17]) Given a permutation $\sigma \in \mathbf{\Sigma}$, for every vector $y \in \mathcal{P}(\sigma)$ the vector h_σ^f is an extreme subgradient of $\hat{f}$ at y. If y belongs to the (strict) interior of $\mathcal{P}(\sigma)$, h_σ^f is a unique subgradient corresponding to $\hat{f}$ at y.*

The above lemma points out a critical fact about the subgradients of the Lovász extension, in that they depend only on the total ordering of a vector and are independent of the vector itself. This also implies that if y is totally ordered (it belongs to the interior of $\mathcal{P}(\sigma_y)$) then $\partial_{\hat{f}}(y)$ consists of a single (unique) subgradient. Hence, two entirely different but identically ordered vectors will have identical extreme subgradients. This fact is important when defining and understanding the properties of the LB divergence.

2.3 The Lovász Bregman divergences

We are now in position to define the Lovász-Bregman divergence. Throughout this paper, we restrict $\text{dom}(\hat{f})$ to be $[0,1]^n$. For the applications we consider in this paper, we lose no generality with this assumption, since the scores can easily be scaled to lie within this volume.

Consider the case when y is totally ordered, and correspondingly $|\mathbf{\Sigma}_y| = 1$. It follows then from Lemma 2.2 that there exists a unique subgradient and $\mathcal{H}_{\hat{f}}(y) = h_{\sigma_y}^f$. Hence for any $x \in [0,1]^n$, we have from Eqn. (6) that [19]:

$$d_{\hat{f}}(x, y) = \hat{f}(x) - \langle x, h_{\sigma_y}^f\rangle = \langle x, h_{\sigma_x}^f - h_{\sigma_y}^f\rangle \quad (8)$$

Notice that this divergence depends only on x and σ_y, and is independent of y itself. In particular, the LB divergence between a vector x and any vector $y \in \mathcal{P}(\sigma)$ is the same for all $y \in \mathcal{P}(\sigma)$ (Lemma 2.2). We also invoke the following lemma from [19]:

Lemma 2.3. *(Theorem 2.2, [19]) Given a submodular function whose polyhedron contains all possible extreme points and x which is totally ordered, $d_{\hat{f}}(x, y) = 0$ if and only if $\sigma_x = \sigma_y$.*

At first sight it seems that the class of submodular functions satisfying Lemma 2.3 is very specific. We point out however that this class is quite general and many instances we consider in this paper belong to this class of functions. For example, it is easy to see that the class of submodular functions $f(X) = g(|X|)$ where g is a concave function satisfying $g(i) - g(i-1) \neq g(j) - g(j-1)$ for $i \neq j$ belong to this class.

Hence the Lovász-Bregman divergence is score based permutation divergence, and we denote it as:

$$d_{\hat{f}}(x||\sigma) = \langle x, h_{\sigma_x}^f - h_\sigma^f\rangle \quad (9)$$

	$f(X)$	$\hat{f}(x)$	$d_{\hat{f}}(x,y)$
1)	$\|X\|\|V\backslash X\|$	$\sum_{i<j} \|x_i - x_j\|$	$\sum_{i<j} \|x_{\sigma(i)} - x_{\sigma(j)}\| I(\sigma_x^{-1}\sigma(i) > \sigma_x^{-1}\sigma(j))$
2)	$g(\|X\|)$	$\sum_{i=1}^{k} x(\sigma_x(i))\delta_g(i)$	$\sum_{i=1}^{n} x(\sigma_x(i))\delta_g(i) - \sum_{i=1}^{k} x(\sigma_y(i))\delta_g(i)$
3)	$\min\{\|X\|, k\}$	$\sum_{i=1}^{k} x(\sigma_x(i))$	$\sum_{i=1}^{k} x(\sigma_x(i)) - \sum_{i=1}^{k} x(\sigma(i))$
4)	$\min\{\|X\|, 1\}$	$\max_i x_i$	$\max_i x_i - x(\sigma(1))$
5)	$\sum_{i=1}^{n} \|I(i \in X) - I(i+1 \in X)$	$\sum_{i=1}^{n} \|x_i - x_{i+1}\|$	$\sum_{i=1}^{n} \|x_i - x_{i+1}\| I(\sigma_x^{-1}\sigma(i) > \sigma_x^{-1}\sigma(i+1))$
6)	$I(1 \leq \|A\| \leq n-1)$	$\max_i x(i) - \min_i x(i)$	$\max_i x(i) - x(\sigma(1)) - \min_i x(i) + x(\sigma(n))$
7)	$I(A \neq \emptyset, A \neq V)$	$\max_{i,j} \|x_i - x_j\|$	$\max_{i,j} \|x_i - x_j\| - \|x(\sigma(1) - x(\sigma(n))\|$

Table 1: Examples of the LB divergences. $I(.)$ is the Indicator fn.

As we shall see in the next section, this divergence has a number of properties akin to the standard permutation based distance metrics. Since a large class of submodular functions satisfy the above property (of having all possible extreme points), the Lovász-Bregman divergence forms a large class of divergences.

The case when y is not totally ordered can be handled similarly [20].

2.4 Lovász Bregman Divergence Examples

Below is a partial list of some instances of the Lovász-Bregman divergence. We shall see that a number of these are closely related to many standard permutation based metrics. Table 1 considers several other examples of LB divergences.

Cut function and symmetric submodular functions: A fundamental submodular function, which is also symmetric, is the graph cut function. This is $f(X) = \sum_{i \in X} \sum_{j \in V \backslash X} d_{ij}$. The Lovász extension of f is $\hat{f}(x) = \sum_{i,j} d_{ij}(x_i - x_j)_+$ [2]. The LB divergence corresponding to $\hat{f}$ then has a nice form:

$$d_{\hat{f}}(x||\sigma) = \sum_{i<j} d_{\sigma(i)\sigma(j)} |x_{\sigma(i)} - x_{\sigma(j)}| I(\sigma_x^{-1}\sigma(i) > \sigma_x^{-1}\sigma(j)) \tag{10}$$

We in addition assume that d is symmetric (i.e., $d_{ij} = d_{ji}, \forall i,j \in V$) and hence f is also symmetric. Indeed a weighted version of Kendall τ can be written as $d_T^w(\sigma, \pi) = \sum_{i,j:i<j} w_{ij} I(\sigma^{-1}\pi(i) > \sigma^{-1}\pi(j))$ and $d_{\hat{f}}(x||\sigma)$ is exactly then a form of $d_T^w(\sigma_x, \sigma)$, with $w_{ij} = d_{\sigma(i)\sigma(j)}|x_{\sigma(i)} - x_{\sigma(j)}|$. Moreover, if $d_{ij} = \frac{1}{|x_i - x_j|}$, we have $d_{\hat{f}}(x||\sigma) = d_T(\sigma_x, \sigma)$. Hence, we recover the Kendall τ for that particular x.

An interesting special case of this is when $f(X) = |X||V \backslash X|$, in which case we get:

$$d_{\hat{f}}(x||\sigma) = \sum_{i<j} |x_{\sigma(i)} - x_{\sigma(j)}| I(\sigma_x^{-1}\sigma(i) > \sigma_x^{-1}\sigma(j)).$$

Cardinality based monotone submodular functions: Another class of submodular functions is $f(X) = g(|X|)$ for some concave function g. This form induces an interesting class of Lovász Bregman divergences. In this case $h_{\sigma_x}^f(\sigma_x(i)) = g(i) - g(i-1)$. Define $\delta_g(i) = g(i) - g(i-1)$, then:

$$d_{\hat{f}}(x||\sigma) = \sum_{i=1}^{n} x[\sigma_x(i)]\delta_g(i) - \sum_{i=1}^{n} x[\sigma(i)]\delta_g(i). \tag{11}$$

Notice that we can start with any δ_g such that $\delta_g(1) \geq \delta_g(2) \geq \cdots \geq \delta_g(n)$, and through this we can obtain the corresponding function g. Consider a specific example, with $\delta_g(i) = n - i$. Then, $d_{\hat{f}}(x||\sigma) = \sum_{i=1}^{n} x[\sigma(i)]i - x[\sigma_x(i)]i = \langle x, \sigma^{-1} - \sigma_x^{-1} \rangle$. This expression looks similar to the Spearman's rule (Eqn. (2)), except for being additionally weighted by x.

We can also extend this in several ways. For example, consider a restriction to the top m elements ($m < n$). Define $f(X) = \min\{g(|X|), g(m)\}$. Then it is not hard to verify that:

$$d_{\hat{f}}(x||\sigma) = \sum_{i=1}^{m} x[\sigma_x(i)]\delta_g(i) - \sum_{i=1}^{m} x[\sigma(i)]\delta_g(i). \tag{12}$$

A specific example is $f(X) = \min\{|X|, m\}$, where

$$d_{\hat{f}}(x||\sigma) = \sum_{i=1}^{m} x(\sigma_x(i)) - x(\sigma(i)). \tag{13}$$

In this case, the divergence between x and σ is the difference between the largest m values of x and the m first values of x under the ordering σ. Here the ordering is not really important, but it is just the sum of the top m values and hence if σ_x and σ, under x, have the same sum of first m values, the divergence is zero (irrespective of their ordering or individual element valuations). We can also define δ_g, such that $\delta_g(1) = 1$ and $\delta_g(i) = 0, \forall i \neq 1$. Then, $d_{\hat{f}}(x||\sigma) = \max_j x(j) - x(\sigma(1))$ (this is equivalent to Eqn. (13) when $m = 1$). In this case, the divergence depends only on the top value, and if σ_x and σ have the same leading element, the divergence is zero.

2.5 Lovász Bregman as ranking measures

In this section, we show how the Lovász Bregman subsumes and is closely related to several commonly used loss functions in Information Retrieval connected to ranking.

The Normalized Discounted Cumulative Gain (NDCG): The NDCG metric [22] is one of the most widely used ranking measures in web search. Given a relevance vector r, where the entry r_i typically provides the relevance of a document $i \in \{1, 2, \cdots, n\}$ to a query, and an ordering of documents σ, the NDCG loss function with respect to a discount function D is defined as:

$$\mathcal{L}(\sigma) = \frac{\sum_{i=1}^{k} r(\sigma_r(i))D(i) - \sum_{i=1}^{k} r(\sigma(i))D(i)}{\sum_{i=1}^{k} r(\sigma_r(i))D(i)} \quad (14)$$

Here $k \leq n$ is often used as a cutoff. Intuitively the NDCG loss compares an ordering σ to the best possible ordering σ_r. The typical choice of $D(i) = \frac{1}{\log(1+i)}$, though in general any decreasing function can be used. This function is closely related to a form of the LB divergence. In particular, notice that $\mathcal{L}(\sigma) \propto \sum_{i=1}^{k} r(\sigma_r(i))D(i) - \sum_{i=1}^{k} r(\sigma(i))D(i)$ (since the denominator of Eqn. (14) is a constant) which is form of Eqn. (12) with $m = k$ and choosing the function $g(i) = \sum_{j=1}^{i} D(i)$.

Area Under the Curve: Another commonly used ranking measure is the Area under the curve [35]. Unlike NDCG however, this just relies on a partial ordering of the documents and not a complete ordering. In particular denote G as a set of "good" documents and B as a set of "bad" documents. Then the loss function $\mathcal{L}(\sigma)$ corresponding to an ordering of documents σ is

$$\mathcal{L}(\sigma) = \frac{1}{|G||B|} \sum_{g \in G, b \in B} I(\sigma(g) > \sigma(b)). \quad (15)$$

This can be seen as an instance of LB divergence corresponding to the cut function by choosing $d_{ij} = \frac{1}{|G||B|}, \forall i, j,\ x_g = 1, \forall g \in G$ and $x_b = 0, \forall b \in B$.

3 Lovász Bregman Properties

In this section, we shall analyze some interesting properties of the LB divergences. While many of these properties show strong similarities with permutation based metrics, the Lovász Bregman divergence enjoys some unique properties, thereby providing novel insight into the problem of combining and clustering ordered vectors.

Non-negativity and convexity: The LB divergence is a divergence, in that $\forall x, \sigma, d_{\hat{f}}(x||\sigma) \geq 0$. Additionally if the submodular polyhedron of f has all possible extreme points, $d_{\hat{f}}(x||\sigma) = 0$ iff $\sigma_x = \sigma$. Also the Lovász-Bregman divergence $d_{\hat{f}}(x||\sigma)$ is convex in x for a given σ.

Equivalence Classes: The LB divergence of submodular functions which differ only in a modular term are equal. Hence for a submodular function f and a modular function m, $d_{\widehat{f+m}}(x||\sigma) = d_{\hat{f}}(x||\sigma)$. Since any submodular function can be expressed as a difference between a polymatroid and a modular function [11], it follows that it suffices to consider polymatroid functions while defining the LB divergences.

Linearity and Linear Separation: The LB divergence is a linear operator in the submodular function f. Hence for two submodular functions $f_1, f_2, d_{\widehat{f_1+f_2}}(x||\sigma) = d_{\hat{f}_1}(x||\sigma) + d_{\hat{f}_2}(x||\sigma)$. The LB divergence has the property of linear separation — the set of points x equidistant to two permutations σ_1 and σ_2 (i.e., $\{x : d_{\hat{f}}(x||\sigma_1) = d_{\hat{f}}(x||\sigma_2)\}$) comprise a hyperplane. Similarly, for any x, the set of points y such that $d_{\hat{f}}(x, y) = \text{constant}$, is $\mathcal{P}(\sigma_y)$.

Invariance over relabelings: The permutation based distance metrics have the property of being left invariant with respect to reorderings, i.e., given permutations π, σ, τ, $d(\pi, \sigma) = d(\tau\pi, \tau\sigma)$.

While this property may not be true of the Lovász Bregman divergences in general, the following theorem shows that this is true for a large class of them.

Theorem 3.1. *Given a submodular function f, such that $\forall \sigma, \tau \in \mathbf{\Sigma}$, $h^f_{\tau\sigma} = \tau h^f_\sigma$, $d_{\hat{f}}(x||\sigma) = d_{\hat{f}}(\tau x||\tau\sigma)$.*

This property seems a little demanding for a submodular function. But a large class of submodular functions can be seen to have this property. In fact, it can be verified that any cardinality based submodular function has this property.

Corollary 3.1.1. *Given a submodular function f such that $f(X) = g(|X|)$ for some function g, then $d_{\hat{f}}(x||\sigma) = d_{\hat{f}}(\tau x, \tau\sigma)$.*

This follows directly from Eqn. (11) and observing that the extreme points of the corresponding polyhedron are reorderings of each other. In other words, in these cases the submodular polyhedron forms a permutahedron. This property is true even for sums of such functions and therefore for many of the special cases which we have considered.

Dependence on the values and not just the orderings: We shall here analyze one key property of the LB divergence that is not present in other permutation based divergences. Consider the problem of combining rankings where, given a collection of scores $x^1, \cdots, x^n$, we want to come up with a joint ranking. An extreme case of this is where for some x all the elements are the same. In this case x expresses no preference in the joint ranking. Indeed it is easy to verify that for such an x, $d_{\hat{f}}(x||\sigma) = 0, \forall\sigma$. Now given a x where all the elements are almost equal (but not exactly equal), even though this vector is totally ordered, it expresses a very low confidence in it's ordering. We would expect for such an x, $d_{\hat{f}}(x||\sigma)$ to be small for every σ. Indeed we have the following result:

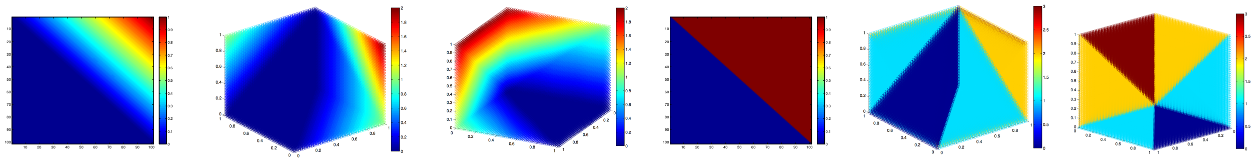

(a) LB2D (b) LB3D1 (c) LB3D2 (d) KT2D (e) KT3D1 (f) KT3D2

Figure 1: A visualization of $d_{\hat{f}}(x||\sigma)$ (left three) and $d_T(\sigma_x, \sigma)$ (right three). The figures shows a visualization in 2D, and two views in 3D for each, with σ as $\{1, 2\}$ and $\{1, 2, 3\}$ and $x \in [0, 1]^2$ and $[0, 1]^3$ respectively.

Theorem 3.2. *Given a monotone submodular function f and any permutation σ,*

$$d_{\hat{f}}(x||\sigma) \leq \epsilon n(\max_j f(j) - \min_j f(j|V\backslash j)) \leq \epsilon n \max_j f(j)$$

where $\epsilon = \max_{i,j} |x_i - x_j|$ and $f(j|A) = f(A \cup j) - f(A)$.

The above theorem implies that if the vector x is such that all it's elements are almost equal, then ϵ is small and the LB divergence is also proportionately small. This bound can be improved in certain cases. For example for the cut function, with $f(X) = |X||V\backslash X|$, we have that $d_{\hat{f}}(x||\sigma) \leq \epsilon d_T(\sigma_x, \sigma) \leq \epsilon n(n-1)/2$, where d_T is the Kendall τ.

Priority for higher rankings: We show yet another nice property of the LB divergence with respect to a natural priority in rankings. This property has to do intrinsically with the submodularity of the generator function. We have the following theorem, that demonstrates this:

Lemma 3.1. *Given permutations σ, π, such that $\mathcal{P}(\sigma)$ and $\mathcal{P}(\pi)$ share a face (say $S_k^\sigma \neq S_k^\pi$) and $x \in \mathcal{P}(\pi)$), then $d_{\hat{f}}(x||\sigma) = (x_k - x_{k+1})(f(\sigma_x(k)|S_{k-1}^\sigma) - f(\sigma_x(k)|S_k^\sigma))$.*

This result directly follows from the definitions. Now consider the class of submodular function f such that $\forall j, k \notin X, j \neq k, f(j|S) - f(j|S \cup k)$ is monotone decreasing as a function of S. An example of such a submodular function is again $f(X) = g(|X|)$, for a concave function g. Then it is clear that from the above Lemma that $d_{\hat{f}}(x||\sigma)$ will be larger for smaller k. In other words, if π and σ differ in the starting of the ranking, the divergence is more than if π and σ differ somewhere towards the end of the ranking. This kind of weighting is more prominent for the class of functions which depend on the cardinality, i.e., $f(X) = g(|X|)$. Recall that many of our special cases belong to this class. Then we have that $d_{\hat{f}}(x||\sigma) = \sum_{i=1}^n \{x(\sigma_x(i)) - x(\sigma(i))\}\delta_g(i)$. Now since $\delta_g(1) \geq \delta_g(2) \geq \cdots \geq \delta_g(n)$, it then follows that if σ_x and σ differ in the start of the ranking, they are penalized more.

Extensions to partial orderings and top m-Lists: So far we considered notions of distances between a score x and a complete permutation σ. Often we may not be interested in a distance to a total ordering σ, but just a distance to a say a top-m list [26] or a partial ordering between elements [38, 8]. The LB divergence also has a nice interpretation for both of these. In particular, in the context of top m lists, we can use Eqn. (12). This exactly corresponds to the divergence between different or possibly overlapping sets of m objects. Moreover, if we are simply interested in the top m elements without the orderings, we have Eqn. (13). A special case of this is when we may just be interested in the top value. Another interesting instance is of partial orderings, where we do not care about the total ordering. For example, in web ranking we often just care about the relevant and irrelevant documents and that the relevant ones should be placed above the irrelevant ones. We can then define a distance $d_{\hat{f}}(x||\mathcal{P})$ where $\mathcal{P}$ refers to a partial ordering by using the cut based Lovász Bregman (Eqn. (10)) and defining the graph to have edges corresponding to the partial ordering. For example if we are interested in a partial order $1 > 2, 3 > 2$ in the elements $\{1, 2, 3, 4\}$, we can define $d_{1,2} = d_{3,2} = 1$ with the rest $d_{ij} = 0$ in Eqn. (10). Defined in this way, the LB divergence then measures the distortion between a vector x and the partial ordering $1 > 2, 3 > 2$. In all these cases, we see that the extensions to partial rankings are natural in our framework, without needing to significantly change the expressions to admit these generalizations.

Lovász-Mallows model: In this section, we extend the notion of Mallows model to the LB divergence. We first define the Mallows model for the LB divergence:

$$p(x|\theta, \sigma) = \frac{\exp(-\theta d_{\hat{f}}(x||\sigma))}{Z(\theta, \sigma)}, \quad \theta \geq 0. \tag{16}$$

For this distribution to be a valid probability distribution, we assume that the domain $\mathcal{D}$ of x to be a bounded set (say for example $[0, 1]^n$). We also assume that the domain is symmetric over permutations (i.e., for all $\sigma \in \mathbf{\Sigma}$, if $x \in \mathcal{D}, x\sigma \in \mathcal{D}$. Unlike the standard Mallow's model, however, this is defined over scores (or valuations) as opposed to permutations.

Given the class of LB divergences defining a probability distribution over such a symmetric set (i.e., the divergences are invariant over relabelings) it follows that

$Z(\theta,\sigma) = Z(\theta)$. The reason for this is:

$$\begin{aligned} Z(\theta,\sigma) &= \int_x \exp(-\theta d_{\hat{f}}(x,\sigma))dx \\ &= \int_x \exp(-\theta d_{\hat{f}}(x\sigma^{-1},\sigma_0))dx \\ &= \int_{x'} \exp(-\theta d_{\hat{f}}(x',\sigma_0))dx' = Z(\theta) \end{aligned}$$

where $\sigma_0 = \{1,2,\cdots,n\}$. We can also define an extended Mallows model for combining rankings, analogous to [27]. Unlike the Mallows model however this is a model over permutations given a collection of vectors $\mathcal{X} = \{x_1,\cdots,x_n\}$ and parameters $\Theta = \{\theta_1,\cdots,\theta_n\}$.

$$p(\sigma|\Theta,\mathcal{X}) = \frac{\exp(-\sum_{i=1}^n \theta_i d_{\hat{f}}(x_i||\sigma))}{Z(\Theta,\mathcal{X})} \tag{17}$$

This model can be used to combine rankings using the LB divergences, in a manner akin to Cranking [27]. This extended Lovász-Mallows model also admits an interesting Bayesian interpretation, thereby providing a generative view to this model:

$$p(\sigma|\Theta,\mathcal{X}) \propto p(\sigma)\prod_{i=1}^n p(x_i|\sigma,\theta_i). \tag{18}$$

Again this directly follows from the fact that in this case, in the Lovász-Mallows model, the normalizing constants (which are independent of σ) cancel out. We shall actually see some very interesting connections between this conditional model and web ranking.

4 Applications

Rank Aggregation: As argued above, the LB divergence is a natural model for the problem of combining scores, where both the ordering and the valuations are provided. If we ignore the values, but just consider the rankings, this then becomes rank aggregation. A natural choice in such problems is the Kendall τ distance [27, 26, 31]. On the other hand, if we consider only the values without explicitly modeling the orderings, then this becomes an incarnation of boosting [16]. The Lovász-Bregman divergence tries to combine both aspects of this problem – by combining orderings using a permutation based divergence, while simultaneously using the additional information of the confidence in the orderings provided by the valuations. We can then pose this problem as:

$$\sigma \in \operatorname*{argmin}_{\sigma' \in \mathbf{\Sigma}} \sum_{i=1}^n d_{\hat{f}}(x^i||\sigma') \tag{19}$$

The above notion of the representative ordering (also known as the mean ordering) is very common in many applications [3] and has also been used in the context of combining rankings [31, 27, 26]. Unfortunately this problem in the context of the permutation based metrics were shown to be NP hard [4]. Surprisingly for the LB divergence this problem is easy (and has a closed form). In particular, the representative permutation is exactly the ordering corresponding to the arithmetic mean of the elements in $\mathcal{X}$.

Lemma 4.1. *[19] Given a submodular function f, the Lovász Bregman representative (Eqn. (19)) is $\sigma = \sigma_\mu$, where $\mu = \frac{1}{n}\sum_{i=1}^n x^i$*

This result builds on the known result for Bregman divergences [3]. This seems somewhat surprising at first. Notice, however, that the arithmetic mean uses additional information about the scores and its confidence, as opposed to just the orderings. In this context, the result then seems reasonable since we would expect that the representatives be closely related to the ordering of the arithmetic mean of the objects. We shall also see that this notion has in fact been ubiquitously but unintentionally used in the web ranking and information retrieval communities.

We illustrate the utility of the Lovász Bregman rank aggregation through the following argument. Assume that a particular vector x is uninformative about the true ordering (i.e, the values of x are almost equal). Then with the LB divergence and any permutation π, $d(x||\pi) \approx 0$, and hence this vector will not contribute to the mean ordering. Instead if we use a permutation based metric, it will ignore the values but consider only the permutation. As a result, the mean ordering tends to consider such vectors x which are uninformative about the true ordering. As an example, consider a set of scores: $\mathcal{X} = \{1.9,2\},\{1.8,2\},\{1.95,2\},\{2,1\},\{2.5,1.2\}$. The representative of this collection as seen by a permutation based metric would be the permutation $\{1,2\}$ though the former three vectors have very low confidence. The arithmetic mean of these vectors is however $\{2.03,1.64\}$ and the Lovász Bregman representative would be $\{2,1\}$.

The arithmetic mean also provides a notion of confidence of the population. In particular, if the total variation [2] of the arithmetic mean is small, it implies that the population is not confident about its ordering, while if the variation is high, it provides a certificate of a homogeneous population. Figure 1 provides a visualization the Lovász-Bregman divergence using the cut function and the Kendall τ metric, visualized in 2 and 3 dimensions respectively. We see the similarity between the two divergences and at the same time, the dependence on the "scores" in the Lovász-Bregman case.

Learning to Rank: We investigate a specific instance of the rank aggregation problem with reference to the problem of "learning to rank." A large class of algorithms have been proposed for this problem – see [28] for a survey on this. A specific class of algorithms for this problem have focused on maximum margin learning using ranking based loss functions (see [38, 8] and references therein). While we have seen that the ranking based losses themselves are instances of the LB divergence, the feature functions are also closely related.

In particular, given a query q, we denote a feature vector corresponding to document $i \in \{1, 2, \cdots, n\}$ as $x_i \in \mathbb{R}^d$, where each element of x_i denotes a quality of document i based on a particular indicator or feature. Denote $\mathcal{X} = \{x_1, \cdots, x_n\}$. We assume we have d feature functions (one might be for example a match with the title, another might be pagerank, etc). Denote x_i^j as the score of the j^{th} feature corresponding to document i and $x^j \in \mathbb{R}^n$ as the score vector corresponding to feature j over all the documents. In other words, $x^j = (x_1^j, x_2^j, \cdots, x_n^j)$. One possible choice of feature function is:

$$\phi(\mathcal{X}, \sigma) = \sum_{j=1}^{d} w_j d_{\hat{f}}(x^j || \sigma) \tag{20}$$

for a weight vector $w \in \mathbb{R}^d$. Given a particular weight vector w, the inference problem then is to find the permutation σ which minimizes $\phi(\mathcal{X}, \sigma)$. Thanks to Lemma 4.1, the permutation σ is exactly the ordering of the vector $\sum_{j=1}^{n} w_j x^j$. It is not hard to see that this exactly corresponds to ordering the scores $w^\top x_i$ for $i \in \{1, 2, \cdots, n\}$. Interestingly many of the feature functions used in [38, 8] are forms closely related Eqn. (20). In fact the motivation to define these feature functions is exactly that the inference problem for a given set of weights w be solved by simply ordering the scores $w^\top x_i$ for every $i \in \{1, 2, \cdots, n\}$ [8]. We see that through Eqn. (20), we have a large class of possible feature functions for this problem.

We also point out a connection between the learning to rank problem and the Lovász-Mallows model. In particular, recent work [12] defined a conditional probability model over permutations as:

$$p(\sigma | w, \mathcal{X}) = \frac{\exp(w^\top \phi(\mathcal{X}, \sigma))}{Z}. \tag{21}$$

This conditional model is then exactly the extended Lovász-Mallows model of Eqn. (17) when ϕ is defined as in Eqn. (20). The conditional models used in [12] are in fact closely related to this and correspondingly Eqn. (21) offers a large class of conditional ranking models for the learning to rank problem.

Clustering: A natural generalization of rank aggregation is the problem of clustering. In this context, we assume a heterogeneous model, where the data is represented as mixtures of ranking models, with each mixture representing a homogeneous population. It is natural to define a clustering objective in such scenarios. Assume a set of representatives $\Sigma = \{\sigma^1, \cdots, \sigma^k\}$ and a set of clusters $\mathcal{C} = \{\mathcal{C}^1, \mathcal{C}^2, \cdots, \mathcal{C}^k\}$. The clustering objective is then: $\min_{\mathcal{C}, \Sigma} \sum_{j=1}^{k} \sum_{i: x_i \in \mathcal{C}_j} d_{\hat{f}}(x_i || \sigma_i)$. As shown in [19], a simple k-means style algorithm finds a local minima of the above objective. Moreover due to simplicity of obtaining the means in this case, this algorithm is extremely scalable and practical.

5 Discussion

To our knowledge, this work is the first introduces the notion of "score based divergences" in preference and ranking based learning. Many of the results in this paper are due to some interesting properties of the Lovász extension and Bregman divergences. This also provides interesting connections between web ranking and the permutation based metrics. This idea is mildly related to the work of [36] where they use the Choquet integral (of which the Lovász extension is a special case) for preference learning. Unlike our paper, however, they do not focus on the divergences formed by the integral. Finally, it will be interesting to use these ideas in real world applications involving rank aggregation, clustering, and learning to rank.

Acknowledgments: We thank Matthai Phillipose, Stefanie Jegelka and the melodi-lab submodular group at UWEE for discussions and the anonymous reviewers for very useful reviews. 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.

References

[1] F. Bach. Structured sparsity-inducing norms through submodular functions. *NIPS*, 2010.

[2] F. Bach. Learning with Submodular functions: A convex Optimization Perspective. *Arxiv*, 2011.

[3] A. Banerjee, S. Meregu, I. S. Dhilon, and J. Ghosh. Clustering with Bregman divergences. *JMLR*, 6:1705–1749, 2005.

[4] J. Bartholdi, C. Tovey, and M. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and welfare*, 6(2):157–165, 1989.

[5] L. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Comput. Math and Math Physics*, 7, 1967.

[6] L. Busse, P. Orbanz, and J. Buhmann. Cluster analysis of heterogeneous rank data. In *In ICML*, volume 227, pages 113–120, 2007.

[7] Y. Censor and S. Zenios. *Parallel optimization: Theory, algorithms, and applications.* Oxford University Press, USA, 1997.

[8] S. Chakrabarti, R. Khanna, U. Sawant, and C. Bhattacharyya. Structured learning for non-smooth ranking losses. In *SIGKDD*, pages 88–96. ACM, 2008.

[9] G. Choquet. Theory of capacities. In *Annales de linstitut Fourier*, volume 5, page 87, 1953.

[10] D. Critchlow. *Metric methods for analyzing partially ranked data in Lecture Notes in Statistics No. 34*. Springer-Verlag, Berlin 1985, 1985.

[11] W. H. Cunningham. Decomposition of submodular functions. *Combinatorica*, 3(1):53–68, 1983.

[12] A. Dubey, J. Machchhar, C. Bhattacharyya, and S. Chakrabarti. Conditional models for non-smooth ranking loss functions. In *ICDM*, pages 129–138, 2009.

[13] J. Edmonds. Submodular functions, matroids and certain polyhedra. *Combinatorial structures and their Applications*, 1970.

[14] M. Fligner and J. Verducci. Distance based ranking models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 359–369, 1986.

[15] M. Fligner and J. Verducci. Multistage ranking models. *Journal of the American Statistical Association*, 83(403):892–901, 1988.

[16] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *JMLR*, 4:933–969, 2003.

[17] S. Fujishige. *Submodular functions and optimization*, volume 58. Elsevier Science, 2005.

[18] G. Gordon. Regret bounds for prediction problems. In *In COLT*, pages 29–40. ACM, 1999.

[19] R. Iyer and J. Bilmes. The submodular Bregman and Lovász-Bregman divergences with applications. In *NIPS*, 2012.

[20] R. Iyer and J. Bilmes. The Lovász-Bregman Divergence and connections to rank aggregation, clustering and web ranking: Extended Version of UAI paper. 2013.

[21] R. Iyer, S. Jegelka, and J. Bilmes. Fast semidifferential based submodular function optimization. In *ICML*, 2013.

[22] K. Järvelin and J. Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In *In SIGIR*, pages 41–48. ACM, 2000.

[23] M. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[24] K. Kirchhoff et al. Combining articulatory and acoustic information for speech recognition in noisy and reverberant environments. In *ICSLP*, volume 98, pages 891–894. Citeseer, 1998.

[25] K. Kiwiel. Proximal minimization methods with generalized Bregman functions. *SIAM Journal on Control and Optimization*, 35(4):1142–1168, 1997.

[26] A. Klementiev, D. Roth, and K. Small. Unsupervised rank aggregation with distance-based models. In *ICML*, 2008.

[27] G. Lebanon and J. Lafferty. Cranking: Combining rankings using conditional probability models on permutations. In *ICML*, 2002.

[28] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[29] L. Lovász. Submodular functions and convexity. *Mathematical Programming*, 1983.

[30] C. Mallows. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130, 1957.

[31] M. Meilă, K. Phadnis, A. Patterson, and J. Bilmes. Consensus ranking under the exponential model. In *In UAI*, 2007.

[32] T. Murphy and D. Martin. Mixtures of distance-based models for ranking data. *Computational statistics & data analysis*, 41(3):645–655, 2003.

[33] R. Rockafellar. *Convex analysis*, volume 28. Princeton Univ Pr, 1970.

[34] A.-V. I. Rosti, N. F. Ayan, B. Xiang, S. Matsoukas, R. Schwartz, and B. Dorr. Combining outputs from multiple machine translation systems. In *NAACL - HLT*, 2007.

[35] K. A. Spackman. Signal detection theory: Valuable tools for evaluating inductive learning. In *Proceedings of the sixth international workshop on Machine learning*, 1989.

[36] A. F. Tehrani, W. Cheng, and E. Hüllermeier. Preference learning using the choquet integral: The case of multipartite ranking. *IEEE Transactions on Fuzzy Systems*, 2012.

[37] M. Telgarsky and S. Dasgupta. Agglomerative Bregman clustering. *In ICML*, 2012.

[38] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *SIGIR*. ACM, 2007.

Solving Limited-Memory Influence Diagrams Using Branch-and-Bound Search

Arindam Khaled and **Eric A. Hansen**
Dept. of Computer Science and Eng.
Mississippi State University
Mississippi State, MS 39762
{ak697, hansen}@cse.msstate.edu

Changhe Yuan
Dept. of Computer and Information Science
Queens College/CUNY
Queens, NY 11367
changhe.yuan@qc.cuny.edu

Abstract

A limited-memory influence diagram (LIMID) generalizes a traditional influence diagram by relaxing the assumptions of regularity and no-forgetting, allowing a wider range of decision problems to be modeled. Algorithms for solving traditional influence diagrams are not easily generalized to solve LIMIDs, however, and only recently have exact algorithms for solving LIMIDs been developed. In this paper, we introduce an exact algorithm for solving LIMIDs that is based on branch-and-bound search. Our approach is related to the approach of solving an influence diagram by converting it to an equivalent decision tree, with the difference that the LIMID is converted to a much smaller decision graph that can be searched more efficiently.

1 Introduction

An influence diagram (ID) is a compact graphical model of a decision problem under uncertainty [5]. In its traditional form, an ID satisfies the assumptions of *regularity* and *no forgetting*, which means that decisions are temporally ordered and each decision is conditioned on all relevant previous observations and decisions. Lauritzen and Nilsson [8] introduced a more general model, called a *limited-memory influence diagram* (LIMID), that allows the regularity and no-forgetting assumptions to be relaxed in order to model a wider range of decision problems. In particular, relaxing the regularity assumption allows modeling of cooperative multi-agent decision problems where one agent is not aware of some or all decisions of another agent. (Note that Howard and Matheson [5] call the regularity assumption the *single decision maker* condition.) Relaxing the no-forgetting assumption allows a decision to be conditioned on a limited number of relevant previous observations and decisions, allowing tradeoffs between the quality of a decision strategy and the complexity of finding it.

Algorithms for solving traditional IDs, such as the join tree algorithm [6], make use of the regularity and no-forgetting assumptions. Thus they cannot be easily generalized to solve LIMIDs. The first algorithm developed to solve LIMIDs, due to Nilsson and Lauritzen [14], is an iterative solution procedure, called single policy updating, that only finds an exact solution under very limited conditions on the structure of the ID; in general, it converges to a locally-optimal solution. The first exact and general algorithm for solving LIMIDs, developed by de Campos and Ji [1], reformulates a LIMID as a credal network inference problem that can be solved by mixed integer programming. Maua and de Campos [11, 10] recently developed a more efficient exact algorithm for solving LIMIDs based on variable elimination, called multiple policy updating.

In this paper, we introduce another exact algorithm for solving LIMIDs. Our approach builds on the work of Yuan et al. [24], who describe a branch-and-bound search algorithm for solving a traditional ID and show that it can outperform other approaches to solving IDs for multi-stage decision problems. We adopt the same branch-and-bound approach, but with some important differences. The branch-and-bound algorithm for solving a traditional ID is a tree-search algorithm in which each path through the decision tree represents perfect memory of a particular history of decisions and observations, in keeping with the no-forgetting assumption of a traditional ID. By contrast, our branch-and-bound algorithm for solving LIMIDs searches in a much smaller search *graph* in which different paths to the same node of the graph represent different histories where the differences are not "remembered." By collapsing the search tree into a smaller search graph in which fewer histories are distinguished, the branch-and-bound approach can solve the optimization problem for LIMIDs much more efficiently. That is, the new graph-search techniques we introduce leverage the opportunities for faster strategy computation provided by the LIMID model. We also develop new techniques for probabilistic inference and bounds computation in the search graph that further enhance this approach to solving LIMIDs. Experimental results demonstrate the effectiveness of this approach.

2 Background

We begin with a review of limited-memory influence diagrams and previous work on solving influence diagrams using branch-and-bound search.

2.1 Influence diagrams

An influence diagram (ID) represents a decision problem by a directed acyclic graph with three types of nodes: chance nodes, decision nodes, and utility nodes. The chance nodes represent random variables, $\mathbf{X} = \{X_1, \ldots, X_n\}$, where each random variable $X_i \in \mathbf{X}$ has an associated domain of possible values, $dom(X_i)$. The decision nodes represent decision variables, $\mathbf{D} = \{D_1, \ldots, D_m\}$, where each decision variable $D_i \in \mathbf{D}$ has an associated domain of possible values, $dom(D_i)$, called actions. For both random variables and decision variables, all domains are assumed to be non-empty and finite. The utility nodes represent (local) utility functions, $\mathbf{U} = \{U_1, \ldots, U_l\}$, that express a decision maker's preferences. By convention, an ID shows chance nodes as circles, decision nodes as squares, and utility nodes as diamonds.

The edges of the graph characterize dependencies among nodes and have a different meaning depending on their destination. Incoming edges to a chance node indicate probabilistic dependence. As in a Bayesian network, each random variable $X_i \in \mathbf{X}$ has an associated conditional probability table $P(X_i|\pi(X_i))$, where $\pi(X_i)$ denotes the set of parent variables of X_i and $dom(\pi(X_i))$ denotes the set of possible instantiations (or states) of the parent variables. Incoming edges into decision nodes are informational, and the parent variables $\pi(D_i)$ of a decision variable $D_i \in \mathbf{D}$, called the *information variables* of the decision, are the variables whose values are known to the decision maker at the time the decision is made. An instantiation of the information variables is called an *information state*, and $dom(\pi(D_i))$ denotes the set of all information states for the decision variable D_i. Incoming edges into utility nodes indicate functional dependence, and a utility function $U_i : \Omega_{\pi(U_i)} \rightarrow \Re$ maps each state of the parent variables $\pi(U_i)$ to a utility value that represents the preference of the decision maker. It is assumed that utility nodes do not have children and the joint utility function $\mathcal{U}$ is additively decomposable such that $\mathcal{U} = \sum_{U_i \in \mathbf{U}} U_i$.

An ID is solved by finding a strategy that maximizes expected utility. A *strategy* $s = \{\delta_{D_i} | D_i \in \mathbf{D}\}$ is a set of policies, one for each decision variable, where a *policy* $\delta_{D_i} : dom(\pi(D_i) \rightarrow dom(D_i)$ is a mapping from the information states of a decision variable to the possible actions for that decision variable. A strategy s induces a joint probability distribution P_s over $\mathbf{X} \cup \mathbf{D}$, as follows,

$$P_s(\mathbf{X} \cup \mathbf{D}) = \Pi_{X_i \in \mathbf{X}} P(X_i|\pi(X_i)) \cdot \Pi_{D_j \in \mathbf{D}} P_s(D_i|\pi(D_i)), \tag{1}$$

where

$$P_s(d|\pi(D_i)) = \begin{cases} 1 & \text{if } \delta_{D_i}(\pi(D_i)) = d, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The expected utility of a *strategy* s is defined as

$$\begin{aligned} EU(s) &= \sum_{c \in (\mathbf{X} \cup \mathbf{D})} P_s(c) \sum_{U_i \in \mathbf{U}} U_i(\pi(U_i)) & (3) \\ &= \sum_{U_i \in \mathbf{U}} \sum_{\pi(U_i)} P_s(\pi(U_i)) \cdot U_i(\pi(U_i)), & (4) \end{aligned}$$

where $c \in (\mathbf{X} \cup \mathbf{D})$ denotes a particular configuration (or instantiation) of the variables of the ID. A strategy s^* is optimal if $EU(s^*) \geq EU(s)$ for all strategies s.

For an ID that satisfies the regularity assumption, the decision variables are temporally ordered. Suppose there are n decision variables $D_1, D_2, ..., D_n$. The decision variables partition the random variables in $\mathbf{X}$ into a collection of disjoint sets $\mathbf{I_0}, \mathbf{I_1}, ..., \mathbf{I_n}$. For each k, where $0 < k < n$, $\mathbf{I_k}$ is the set of random variables that must be observed between D_k and D_{k+1}. $\mathbf{I_0}$ is the set of initial evidence variables that must be observed before D_1. $\mathbf{I_n}$ is the set of variables left unobserved when decision D_n is made. Therefore, a partial order $\prec$ is defined on the ID over $\mathbf{X} \cup \mathbf{D}$, as follows:

$$\mathbf{I_0} \prec D_1 \prec \mathbf{I_1} \prec ... \prec D_n \prec \mathbf{I_n}. \tag{5}$$

When the *no-forgetting* assumption is satisfied, all information variables of earlier decisions are also information variables of later decisions. We call these past information variables the *history*, and, for convenience, we assume that there are *explicit* edges (called information arcs) from history information variables to decision variables. As the number of variables in the history grows, however, the domain of the policy for each decision variable increases exponentially. Methods for structural analysis of relevance have been developed that can distinguish *requisite* observations from those that are irrelevant, and remove information arcs that are not necessary for computation of the optimal strategy [21, 13, 8]. This preprocessing step can be performed prior to any numerical evaluation of the LIMID.

Even if information arcs from irrelevant variables are removed and only relevant information variables are considered for each decision, the domain of the policy for a decision variable may grow exponentially in the number of relevant information variables in the history. Limited-memory influence diagrams [8] address this problem by allowing information arcs from relevant variables to be removed. When an information variable for an earlier decision is not an information variable for a later decision, it means the no-forgetting assumption is violated. If there is not an information arc from an earlier decision variable to a later decision variable, it means the regularity assumption is violated. We use the following example to illustrate the properties of a LIMID.

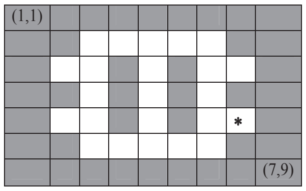


Figure 1: A maze with goal state marked by starred cell.

2.2 Example

Consider a maze navigation problem that can be modeled as an ID [4, 14, 24]. In the maze shown in Figure 1, white cells represent spaces where navigation is possible, and shaded cells represent walls. A robot is initially placed in one of the white cells and its objective is to reach the goal state marked by a star. At each stage, the robot can choose to move to any one of its neighboring cells, or it can stay in its current location. The effect of the robot's attempts to move are stochastic. It moves to the intended neighboring cell with a probability of 0.89 and fails to move with probability 0.089. The probability of sideways movement is 0.01 in each of two possible directions, and the probability of backward movement is 0.001. A move towards a wall has a probability of zero to succeed, and the remaining non-zero probabilities are normalized. The robot's sensors provide incomplete information about the robot's location. They detect neighboring walls, but since more than one location can share the same configuration of neighboring walls, observations do not unambiguously identify the current location. The expected utility received by the robot corresponds to the probability of successfully reaching the goal state by the final stage of the problem. If the robot is in the goal state at the final stage, it receives a utility value of 1; otherwise, it receives a value of 0.

The ID shown in Figure 2(a) represents a two-stage version of the maze navigation problem. The random variables x_i and y_i represent the coordinates of the location of the robot at stage i. The random variables $\{ns_i, es_i, ss_i, ws_i\}$ are the sensor readings in four directions at stage i. The decision variable d_i represents one of the possible actions taken by the robot at that stage. The ID shown in Figure 2(a) is a traditional ID that satisfies the regularity and no-forgetting assumptions. Figure 2(b) shows a LIMID for which the no-forgetting assumption is not satisfied. In this case, a decision is conditioned on all past decisions as well as the present states of the information variables when the decision is made, but not on the information variables for any previous stages. In other words, the robot makes decisions based on its current sensor readings only, without considering any previous sensor readings. Although the IDs shown in Figure 2 represent two-stage decision problems, they are easily extended for any finite number of stages.

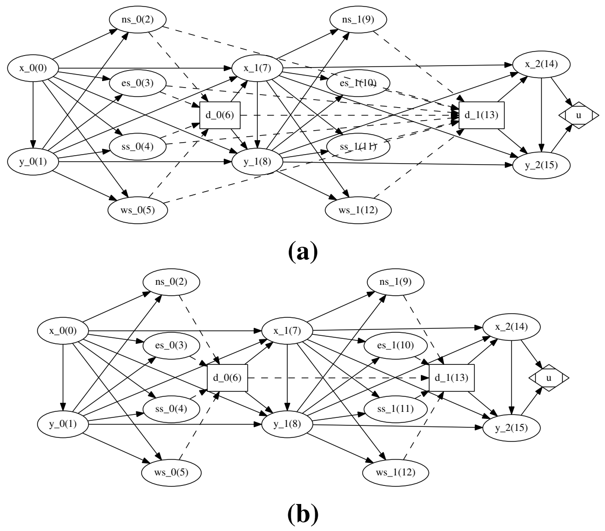


Figure 2: (a) An influence diagram with no-forgetting and (b) a LIMID, both for the maze navigation problem.

2.3 Branch-and-bound solution method

Several exact methods have been developed to solve IDs that satisfy the regularity and no-forgetting assumptions. IDs were first solved by converting them to a decision tree [5]. Subsequently, methods were developed that solve an ID directly [19], or by converting it to some other graphical form, such as a junction tree [20, 6]. For an ID that has been converted to a decision tree, the traditional solution method is the "average-out and fold-back" algorithm [17]. However, improved performance can be achieved using a search algorithm that traverses the decision tree beginning from the root and prunes branches with zero probability [16, 9]. The performance of this approach can be improved further by using bounds to prune the tree, and the results of this paper build on our earlier work on solving IDs using depth-first branch-and-bound search [24]. In this approach, the decision tree corresponds to an AND/OR search tree in which AND nodes correspond to information variables (i.e., chance nodes that have informational arcs into a decision node), OR nodes correspond to decision nodes, and leaf nodes correspond to utility nodes. A path from the root of the search tree to a leaf node corresponds to an instantiation of the information and decision variables of the ID. When traversed by a depth-first search algorithm, the tree is generated "on the fly" and only part of the AND/OR tree needs to be in memory at any one time.

Two issues must be addressed to develop an effective branch-and-bound algorithm. We need bounds to prune the search tree and we need an efficient method for computing posterior probabilities. As we discuss next, our approach to both issues involves construction of a secondary ID we call a *relaxed influence diagram*.

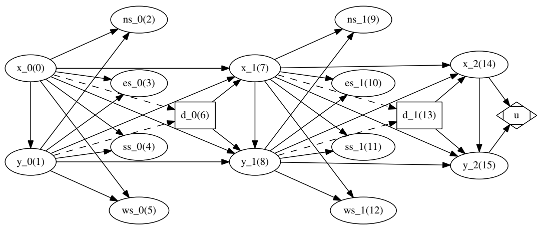


Figure 3: Relaxed ID for two-stage maze problem.

Bounds computation We need bounds on the values of OR nodes in order to prune the AND/OR search tree. Assuming that we are maximizing expected utility, the best value computed so far for any branch of an OR node (i..e, for any action) serves as a lower bound on the optimal value of the OR node. To compute upper bounds, we adopt an approach proposed by Nilsson and Höhle [14]. In this approach, upper bounds are computed by solving a secondary ID that is usually much easier to solve, which we call a relaxed ID. The relaxed ID is created by adding information variables to the original ID that provide the decision maker with additional information (ensuring that the solution of the relaxed ID is an upper bound on the solution of the original ID), while allowing the ID to be simplified by removing *non-requisite* arcs (ensuring that it is much easier to solve than the original ID). Recall that an information arc is *non-requisite* [8, 13] for a decision node D if

$$I_i \perp (\mathbf{U} \cap d(D)) | D \cup (\pi(D) \setminus \{I_i\}), \tag{6}$$

where $d(D)$ and $\pi(D)$ are the descendants and parents of D, respectively, and $\perp$ denotes conditional independence. A *reduction* of an ID is obtained by deleting all *non-requisite* information arcs [14]. Ideally, we want to add information variables that make some or all of the information arcs for each decision node *non-requisite*, and also make the ID easier to solve. The optimal policy for a decision variable, D_j, depends on a set of information variables, N_j, or else it is history-independent. This set N_j can be described as the current state of the decision problem, such that if the decision maker is informed of this state, it does not need to know of any past history to find an optimal policy; in this respect, it fulfills the Markov property and can be said to provide perfect information. N_j is called a *sufficient information set* (SIS) of D_j [24].

Thus the relaxed ID is created in two steps. First, the SIS for each decision is computed in reverse time order, $D_n, ..., D_1$, making each SIS the information variables for its corresponding decision variable. Second, *non-requisite* arcs are removed from the ID. Consider the LIMID in Figure 2(b) as an example. The SIS for d_1 is found as $\{x_1, y_1\}$. The SIS for d_0 is computed to be $\{x_0, y_0\}$. By making $\{x_1, y_1\}$ and $\{x_0, y_0\}$ information variables for d_1 and d_0, respectively, and after removing the non-requisite arcs, a relaxed ID is obtained which is shown in Figure 3.

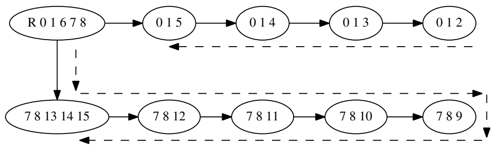


Figure 4: Strong join tree for the relaxed ID in Figure 3.

Incremental join tree evaluation The join tree algorithm, an efficient method for probabilistic inference in a Bayesian network, also provides an efficient method for solving an ID [6]. For optimization problems that can be solved by depth-first branch-and-bound search, Yuan and Hansen [23] describe an incremental version of the join tree algorithm. First developed for a branch-and-bound algorithm for solving the MAP problem, it is used in the branch-and-bound algorithm for solving traditional IDs [24], and we will use it to extend the branch-and-bound approach to LIMIDs. It assumes a static ordering of the variables to be instantiated, and leverages the observation that when only one new variable is instantiated at a time during forward traversal of a branch of a search tree, it is only necessary to perform message passing once along this path in the join tree, broken into separate steps for each instantiating variable. To allow efficient backtracking, the clique and separator potentials of the join tree that are changed during forward traversal are cached in the order that they are changed. During backtracking, the cached potentials can be used to efficiently restore the join tree to its previous state.

A strong join tree constructed from the relaxed ID is used to compute both probabilities and upper bounds for the AND and OR nodes in the search graph. Figure 4 shows a strong join tree for the relaxed ID of Figure 3. We select an order of variable elimination from the join tree that is an extension of the elimination ordering of the ID. Note that one of the partial orders for the LIMID shown in Figure 2(b) is

$$\begin{aligned}&\{ns_0, es_0, ss_0, ws_0\} \prec \{d_0\} \prec \{ns_1, es_1, ss_1, ws_1\}\\ &\prec \{d_1\} \prec \{x_0, y_0, x_1, y_1, x_2, y_2, u\},\end{aligned} \tag{7}$$

Any order of variable expansion for the join tree that satisfies the constraints in Equation 7 can be used for *incremental join tree evaluation*. The order suggests which variable to expand/instantiate next. For example, after ns_0 is expanded, the only message that needs to be sent to obtain $P(es_0|ns_0)$ is the message from clique $(0, 1, 2)$ to $(0, 1, 3)$. If the following order for the maze problem is selected

$$\begin{aligned}&ns_0, es_0, ss_0, ws_0, d_0, ns_1, es_1, ss_1, ws_1, d_1, x_0, y_0,\\ &x_1, y_1, x_2, y_2\end{aligned} \tag{8}$$

then an incremental message-passing scheme can be used in the direction of the dashed arc in Figure 4 in one downward pass of the depth-first search to compute probabilities and upper bounds.

3 AND/OR search graph

We next describe how to formulate the problem of solving a LIMID as an AND/OR graph search problem. There are two main differences between the depth-first-branch-and-bound (DFBnB) algorithm for solving LIMIDs that we develop in the rest of the paper, and the DFBnB algorithm for solving traditional IDs. The first difference is that a LIMID is solved by searching in an AND/OR graph instead of an AND/OR tree. Since a decision maker with limited memory is not able to distinguish all histories, the search space for solving a LIMID is a graph in which different paths that represent different histories can lead to the same OR node because the differences between the histories are not remembered. A second difference is that the message-passing scheme used by the incremental join tree algorithm to compute bounds and probabilities requires some revisions for search in an AND/OR graph instead of an AND/OR tree. We discuss the first difference in Section 3.1 and the second in Section 3.2.

First we review some basic concepts about the AND/OR search space for the decision problem represented by an ID. Recall that AND nodes represent random variables and OR nodes represent decision variables. Any arc emitting from an AND node has a probability attached to it; the sum of all the probabilities associated with the arcs of an AND node is 1.0. Each arc emitting from an OR node represents a decision alternative. The leaf nodes of the search graph have utility values attached to them that are derived from the utility nodes of the ID. The *valuation function* for each node is defined as follows: (a) for a leaf node, the value is its utility value, (b) for an AND node, the value is computed by multiplying the probability associated with each outgoing arc by the utility value of the corresponding child node and then summing these values, and (c) for an OR node, the value is the maximum of the utility values of the child nodes. We use this valuation function to determine the optimal strategy for an ID. We represent a strategy for an ID as a *strategy graph*, which is a subgraph of an AND/OR graph that is defined as follows: (a) it consists of the root of the AND/OR graph; (b) if a non-terminal AND node is in the strategy graph, all its children are in the strategy graph; and (c) if a non-terminal OR node is in the strategy graph, exactly one of its children is in the strategy graph. Given an AND/OR graph that represents all possible histories and strategies for an ID, the decision problem is solved by finding a strategy graph with the maximum value at the root, where the value of the strategy graph is computed based on the valuation function.

The optimal strategy for an ID can always be found by searching in an AND/OR tree. But the tree representation of the problem is inefficient if it contains many repeated subtrees that represent the same subproblem. For a LIMID, the number of repeated subtrees will be much greater than for a traditional ID because many different histories will lead to the same subproblem in which a decision must be made based on limited information that represents only part of the history. In the following, we describe how to convert an AND/OR tree representation of the decision problem for a LIMID into an equivalent AND/OR graph by merging OR nodes that are generated from different histories but represent the same subproblem.

In our approach, we (slightly) limit the class of LIMIDs we consider by assuming the following: if a decision maker is aware of the value of an information variable and then "forgets" it, it cannot recall the forgotten information later on. We call this assumption the *no-recalling-forgotten-information* rule. It is difficult to imagine any realistic decision problem that violates this assumption. But we make the assumption explicit because the LIMID model does not rule out such cases. The assumption simplifies our approach to recognizing and merging equivalent OR nodes.

We also modify the usual definition of an AND/OR graph by introducing a special kind of AND node that allows multiple decisions to be considered in parallel if they represent scenarios that are conditionally independent given that some variables have already been observed. We call this new kind of node a *special AND* (SAND) node. We discuss SAND nodes further in Section 3.1.3.

3.1 Context-based merging of OR nodes

To construct an AND/OR graph instead of an AND/OR tree for solving LIMIDs, the main idea is to merge multiple OR nodes (i.e., decision nodes) that represent the same decision scenario into a single OR node. Our approach uses the concept of the context of an OR node. The *context* of an OR node is defined as the joint state of the information variables remembered by the current decision variable along with the states of previously observed decisions that will influence the descendant utilities of this decision. More formally, the context of an OR node is a set $C_{D_i} = S_{I_{i-1}} \cup S_{D_{i-1}} \cup S_{D_{u(i-1)}}$, where $S_{I_{i-1}}$, $S_{D_{i-1}}$, and $S_{D_{u(i-1)}}$, respectively are the sets of states of the random and decision variables remembered by D_i, and the set of states of the previously expanded decision variables that will influence the descendent utilities of D_i.

For example, Figure 5 shows a partial AND/OR graph for the LIMID in Figure 2. Note that the AND/OR graph is condensed for ease of illustration as the individual random variables for the sensors are grouped together into one AND layer. In our actual AND/OR graph, they correspond to multiple AND layers. The context of the bottom-right OR node in the Figure is $\{1, 1, 1, 1, 0\}$, where $\{1, 1, 1, 1\}$ is the set of present information states, and $\{0\}$ is the last action taken. The previous sensor readings are totally forgotten. That is why the two paths starting from the root converge to this OR node.

In our context-based approach to merging OR nodes, we distinguish three cases; sequential decisions, cooperative decisions, and conditionally-independent decisions. We explain the differences below.

3.1.1 Sequential decisions

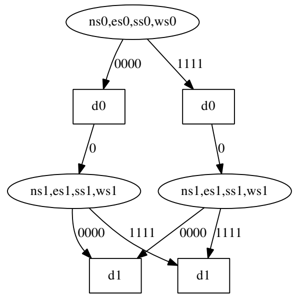


Figure 5: Partial AND/OR graph with merged OR nodes.

A pair of decisions is *sequential* if there is an obvious temporal ordering between them. If all decision pairs in a LIMID are sequential then the elimination order of Equation 5 applies. Figure 2(b) shows an example of a LIMID where all decision pairs are sequential. Duplicate OR nodes are detected using the definition of context provided above and illustrated by the example in Figure 5. Contexts are rather straightforward for sequential decisions given the no-recalling-forgotten-information assumption. Without this assumption, much more complex book-keeping would be needed to keep track of contexts during the search.

Note that in Figure 5, the AND nodes contain multiple random variables to simplify the picture. In the AND/OR search graph created by our implementation, each random variable is represented by a separate AND node, and they are expanded one at a time by the search algorithm.

3.1.2 Cooperative decisions

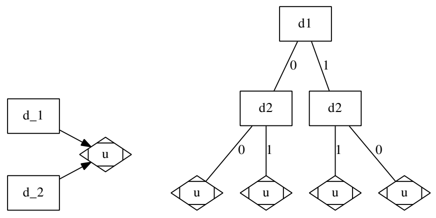


Figure 6: A LIMID with a cooperative-decision pair (left), and its corresponding AND/OR graph (right).

We next consider the case where multiple, simultaneous decisions need to be considered to evaluate a value function. We call this case a *cooperative decision* because the decision makers need to cooperate to select the combination of actions that results in an optimal utility for every possible decision scenario. Unlike the sequential decision scenario, there exists no obvious elimination ordering between decision pairs in this case. Figure 6 shows an example of a cooperative decision pair. In this example, if d_2 is realized after d_1 in the elimination order during the generation of the AND/OR graph, then d_1 needs to be included in d_2's context. The AND/OR graph for this LIMID is also shown in Figure 6. Note that all the OR nodes in this graph are unique.

3.1.3 Conditionally-independent decisions

It is possible that multiple decisions can be expanded in parallel given that some variables have been realized. An example is shown in Figure 7. In this example, the sets of decisions $\{D_1\}$, and $\{D_2\}$ can be expanded in parallel given that the set of variables(s), $\{a\}$, has been observed. We call this situation a *conditionally-independent decision scenario* (CIDS). A CIDS can be determined from the LIMID itself or the strong join tree of the LIMID [7, 21].

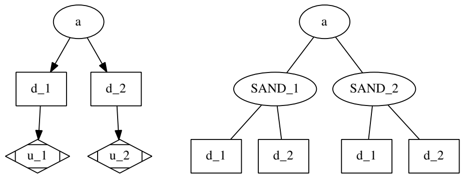


Figure 7: A LIMID with CIDS(left), and its corresponding (partial) AND/OR graph (right).

To solve a LIMID that includes a CIDS, we introduce a new type of node in the AND/OR graph, called a *special AND* or SAND node. When it is realized that multiple decisions can be made in parallel after some nodes in the search graph have been expanded, a SAND node is introduced (in the search graph). A SAND node is different from a regular AND node in two ways: the total number of children for a SAND node is the number of sets of decisions that can be expanded in parallel, and the weight attached to each arc is 1.0. Unlike the regular AND/OR graph search where the rest of the unexpanded nodes are considered for expansion once a node (AND or OR) is expanded for each branch of the AND node, each branch of the SAND node will expand a subset of unexplored nodes for a LIMID with CIDS(s). These subsets are determined by the elimination order of decisions presented by the join tree: each subset will contain the set of decisions and their respective information variables that need to be expanded in sequence in the future. For example, for the LIMID shown in Figure 7, once variable a is expanded, a SAND node, denoted $SAND_1$, will be expanded. This node will have two branches – in one branch the set of decision(s), $\{D_1\}$, will be expanded; the set $\{D_2\}$ will be expanded in the other branch, as illustrated in Figure 7.

3.1.4 Implementation

So far, we have given rules for merging OR nodes but have not discussed their implementation. For each OR node in the graph, we store both its context and utility value (once it is calculated). When an OR node is ready to be generated, its context is calculated and then checked against the contexts of existing OR nodes (which we store in a hash table, described below). If a previously-generated OR node has the same context, the new OR node is not generated. Instead, the existing OR node receives an additional arc from the parent of the new OR node.

For duplicate checking, all generated OR nodes are stored in a hash table indexed by their context. (In cases in which we can decompose the decision problem into well-defined stages, there can be a separate hash table for each stage of the problem.) In our implementation, the context is represented by a string that contains the states of the variables of that decision's context, concatenated by commas. As the search progresses, the context string for each decision node can grow and potentially slow the duplication detection process.

3.1.5 Discussion

The concept of solving an ID by searching in an AND/OR graph, instead of an AND/OR tree, is not new. In the literature on IDs, the process of converting a decision tree to an equivalent graph in which identical subtrees are merged is referred to as *coalescence* [15, 22]. Automating coalescence in the decision tree framework is considered difficult and computationally expensive, however, and solutions are sometimes hand-crafted. A context-based approach to merging OR nodes has been proposed before, for probabilistic inference in Bayesian networks [2] and solving an ID [9]. The primary difference between our approach and previous work is that our approach applies to LIMIDs.

Note that our approach to context-based merging does not necessarily produce the most concise AND/OR graph. A more concise graph could be found by directly comparing probability distributions to detect duplicate OR nodes. We define R_{D_i} as the set of variables that is considered for expansion once the information variables, I_{i-1}, for a decision variable, D_i, are expanded. Formally, $R_{D_i} = D_i \cup I_i \cup D_{i+1} \cup ... \cup I_{n-1} \cup D_n$. Once the information variable set, I_{i-1}, is expanded, the probability distribution of R_{D_i} given D_i's context, $\mathbb{P}(R_{D_i}|C_{D_i})$, could be used to detect duplicate decision scenarios. If multiple decision scenarios share the same distribution then these OR nodes can be collapsed into a single OR node, since they share a subgraph [22]. Although comparing probability distributions in order to merge nodes could generate a more compact AND/OR search graph, it is computationally expensive, and an approach that relies on context-based merging appears to be more practical.

3.2 Incremental probabilities and bounds

We next consider how to modify the incremental join tree algorithm to compute the probabilities and bounds needed by the AND/OR graph search algorithm for solving LIMIDs. Consider the LIMID shown in Figure 2.2(b) as an example, and the relaxed LIMID of Figure 4. We can use the join tree of the relaxed LIMID to compute both the probabilities and bounds needed for the AND/OR graph search. (We can use it to compute probabilities for the AND nodes of the search graph because the same set of actions transforms both the original LIMID and the relaxed LIMID into the same Bayesian network. Adding information arcs to create a relaxed ID only changes the expected utility of the network.) Note that the join tree does not have a clique that contains all four variables; they are in different cliques. Thus we consider the expansion of these variables one by one, generating an AND/OR graph with four layers of AND nodes followed by a layer of OR nodes, as shown in Figure 5.

The AND/OR search graph is generated on-the-fly during the branch-and-bound search. To make this process efficient, we follow the elimination order given by Equation 8 to generate the nodes of the search graph and calculate probabilities. The probabilities for the AND nodes corresponding to variables $\{ns_0, es_0, ss_0, ws_0\}$ can be calculated by sending messages in the following order of cliques: $(0,1,2)$, $(0,1,3)$, $(0,1,4)$, and $(0,1,5)$. Given limited memory, however, calculation of probabilities for the information variables of the next decision needs to be modified.

When a decision stage is considered for expansion, we consider whether the decision variable, D_i, recalls anything from the past. If it does, then the cliques hosting these recalled variables along with the clique, clq_0, that hosts the first information variable for D_i are identified. Then we devise a message-passing scheme that sets evidence for the cliques of the recalled variables and passes messages towards the clique, clq_0. To perform these message propagations, we use a set of temporary potentials, one assigned for each clique and separator, which are initialized to the clique potentials obtained from the initial collection and distribution process of the join tree at the beginning of this process. For our example, when the first information variable, ns_1, for d_1 is about to be expanded, we set evidence to the clique $(0,1,6,7,8)$ with the current state of d_0, and pass in the direction of the clique $(7,8,9)$. Once the clique $(7,8,9)$ receives this message, it sets its current potential to this newly obtained potential. The rest of the information variable expansion process follows the incremental join tree evaluation method proposed in [24]. When backtracking from a decision, we backtrack to the clique hosting the last information variable expanded for the previous expanded decision. The space requirement for our join tree evaluation approach is $O(N)$ if N is the space required for the evaluation approach proposed in [24].

After the AND nodes of $\{ns_0, es_0, ss_0, ws_0\}$ are generated for this example, we need to generate the OR node d_0 and corresponding upper bounds. The subset of information variables for a decision d_i is used to compute the upper bound for d_i. In our example, we do not need to set the states of $\{ns_0, es_0, ss_0, ws_0\}$ as evidence during the computation of expected utility values for d_0 because the states of the information variables for d_0 are not remembered for future decisions. In summary, the basic idea for handling information forgetting in a LIMID is to only send a message to a future decision for calculating probabilities and utility values if the message contains information variables that are remembered by the future decision.

3.3 Optimality of the algorithm

It is not difficult to prove that our AND/OR graph search algorithm finds an optimal strategy. If we do not merge OR nodes, we have an AND/OR tree, and so we first show that the strategy found by an AND/OR tree search is optimal.

Lemma 1 *A DFBnB AND/OR tree search algorithm finds an optimal strategy for a LIMID.*

Proof: Since all the possible policies for each decision node are examined by the search, it must converge to an optimal strategy once the search ends. □

We then argue that merging OR nodes preserves optimality.

Theorem 1 *A DFBnB AND/OR graph search algorithm finds an optimal strategy for a LIMID.*

Proof: Since the subgraphs below any OR nodes that represent the same decision scenario are identical and have the same utility, merging the OR nodes preserves optimality. □

4 Experimental evaluation

We tested the performance of our algorithm in solving the maze problem described in Section 2.2, as well as a classic finite-horizon DEC-POMDP, and several randomly-generated LIMIDs. Experiments were performed on a Windows PC with a Pentium i3 processor and 3GB of RAM.

Tables 1, 2 and 3 show all results in the same format. The column labeled "(d, c, u)" gives the number of decision nodes, chance nodes, and utility nodes, respectively, in the LIMID, and the column labeled "SG" gives the size of the optimal strategy graph measured as the total number of AND and OR nodes it contains. The remaining columns measure the efficiency of the search algorithm. The column labeled "Pruned" gives the number of times a branch of the AND/OR graph was pruned in solving the LIMID, the column labeled "Merged" gives the number of times two OR nodes were merged, and the column labeled "Time" gives the time needed to solve the problem.

(d, c, u)	SG	Pruned	Merged	Time
$(2, 14, 1)$	495	124	528	$109ms$
$(3, 20, 1)$	951	364	2112	$421ms$
$(4, 26, 1)$	1407	688	4224	$952ms$
$(5, 32, 1)$	$1,863$	$2,848$	$18,480$	$4s$
$(6, 38, 1)$	$2,319$	$9,412$	$61,776$	$17s$
$(7, 44, 1)$	$2,775$	$25,768$	$168,960$	$53s$
$(8, 50, 1)$	$3,231$	$90,400$	$593,472$	$3m30s$
$(9, 56, 1)$	$3,687$	$309,184$	$2,027,520$	$13m21s$
$(10, 62, 1)$	$4,143$	$1,058,908$	$6,939,504$	$50m17s$

Table 1: Results for maze navigation LIMID.

4.1 Maze navigation

Table 1 shows results for the maze navigation problem of Section 2.2 when the number of stages is varied from two through ten. The LIMIDs for this problem satisfy the regularity assumption (there is one decision node per stage), but not the no-forgetting assumption (observations are not remembered after the current stage). Of the total number of branches pruned, approximately 40% are pruned based on bounds; the remaining 60% are pruned because the probability of the branch is zero.

4.2 Multi-agent tiger behind door

Table 2 shows results for a finite-horizon DEC-POMDP that represents cooperative multiagent decision making under uncertainty [12]. In this problem, there are two doors, one on the left and one on the right, and two agents. Behind one door is a tiger and behind the other is treasure. Each agent has a choice of three actions: it can open a door on the left or right, or it can listen for the tiger. If an agent hears the tiger behind one of the doors, the tiger is actually there with probability 0.85. At each stage, the agents must each choose an action without knowing what the other agent will choose; thus the regularity assumption is not satisfied. Each agent remembers its previous actions and observations, but is unaware of the other agent's observations. The agents receive better rewards if they coordinate their actions: the reward for opening the door with treasure is greater is both agents open the door together, and the penalty for opening the door with the tiger is less severe if they open that door together. There is a small cost for the listen action.

(d,c,u)	SG	Pruned	Merged	Time
$(4, 6, 2)$	15	12	6	$15ms$
$(6, 9, 3)$	39	35	24	$62ms$
$(8, 12, 4)$	87	737	576	$530ms$
$(10, 15, 5)$	351	$9,529$	$6,984$	$7s$
$(12, 18, 6)$	$1,599$	$42,559$	$31,602$	$49s$
$(14, 21, 7)$	$4,047$	$288,516$	$214,170$	$5m31s$
$(16, 24, 8)$	$10,023$	$2,328,571$	$1,763,904$	$53m21s$

Table 2: Results for multi-agent tiger LIMID.

Table 2 shows results for this problem when the number of stages is varied from two through eight. Our algorithm solves the problem optimally for eight stages in less than an hour. The provably optimal solution reported in [18] is for up to four stages, found by dynamic programming [3]. For this search problem, there are no zero-probability branches. All branches are pruned based on bounds.

4.3 Randomly-generated LIMIDs

We also tested our algorithm on a set of randomly-generated LIMIDs. The LIMIDs were created to have between 10 and 20 stages, with one decision node, one utility node, and between 3 and 6 chance nodes per stage. For each stage, half (and at least 2) of the nodes are selected to be information variables of the decision variable. The utility function for each stage is a function of the decision variable and two randomly-selected random variables from that stage, and potentially the decision variable from the previous stage. Once nodes are generated for all stages, we generate additional informational arcs as follows. For the decision variable of each stage k beginning from the second stage and continuing until the last stage, we add informational arcs from half (and at least 2) of the information variables of the previous stage, selected randomly. This method of adding informational arcs ensures that the no-recalling-forgotten-information rule is satisfied. When adding arcs, we make sure that chance nodes with no children become the information variables for the decision node first so that there are no barren nodes. Each random and decision variable has from 2 to 4 states, and the probabilities for the random nodes are assigned from an uniform probability distribution. The utility values range from -20 to 20.

Table 3 only shows results for a selection of LIMIDs for which the treewidth of the relaxed LIMID does not exceed 12. The LIMIDs solved are larger than those for the maze and tiger problems. For these randomly-generated LIMIDs, not all previous actions are remembered, which appears to make both duplicate detection and probabilistic inference using the incremental join tree algorithm faster.

(d,c,u)	SG	Pruned	Merged	Time
$(10, 46, 10)$	$33,463$	$8,538$	$167,154$	$34s$
$(10, 44, 10)$	$14,785$	$1,475$	$17,817$	$4s$
$(11, 47, 11)$	$16,866$	$6,501$	$153,436$	$50s$
$(12, 60, 12)$	$27,318$	$14,333$	$111,629$	$33s$
$(13, 64, 13)$	$32,087$	$7,220$	$103,560$	$2m34s$
$(13, 60, 13)$	$39,052$	$11,958$	$1,231,516$	$16m26s$
$(13, 65, 13)$	$36,040$	$11,366$	$183,664$	$49s$
$(15, 70, 15)$	$28,080$	$14,546$	$997,728$	$3m26s$
$(16, 72, 16)$	$31,525$	$25,736$	$1,023,012$	$6m13s$
$(18, 84, 18)$	$50,306$	$21,582$	$524,416$	$4m5s$
$(19, 88, 19)$	$130,168$	$7,286$	$140,952$	$46s$
$(20, 84, 20)$	$25,012$	$16,147$	$232,175$	$1m12s$

Table 3: Results for randomly-generated LIMIDs.

4.4 Comparison to variable elimination

The state-of-the-art exact algorithm for solving LIMIDs is a recently-developed variable elimination algorithm called Multiple Policy Updating (MPU) [11, 10]. Although it is not possible to draw definite conclusions about relative performance without direct comparison, we can make some general comments. Like our branch-and-bound algorithm, the MPU algorithm avoids solving redundant decision scenarios by caching and reusing intermediate results. Our algorithm also uses bounds to prune decision scenarios before they are evaluated, however, and can prune zero-probability branches that represent impossible scenarios, and that may give it some advantage, similar to the advantage that the depth-first branch-and-bound algorithm for solving traditional IDs has over other ID algorithms [24]. In reporting results for their variable elimination algorithm, Maua et al. [11, 10] report that it solves randomly-generated LIMIDs with up to 10^{64} strategies and a treewidth bounded by 10. Our branch-and-bound algorithm solves randomly-generated LIMIDs with up to 10^{152} strategies and a treewidth of up to 27. (The multi-agent tiger LIMID has 10^{88} possible strategies and a treewidth of 38. The 10-stage maze problem has 10^{156} possible strategies. The 7-stage maze LIMID has a treewidth of 31; we could not compute the treewidth for more stages than that.) Whereas the scalability of the MPU algorithm is limited by the treewidth of the LIMID, the scalability of the branch-and-bound algorithm appears to be limited by the (usually smaller) treewidth of the relaxed LIMID, which is used to compute bounds and probabilities. In future work, we hope to better characterize the relative performance of these two approaches.

5 Conclusion

We have described a branch-and-bound AND/OR graph search algorithm that finds optimal strategies for LIMIDs, building on earlier work on solving traditional IDs using branch-and-bound search. The approach is especially effective for IDs that represent multistage decision problems.

The branch-and-bound approach performs well even though the bounds used in our implementation are quite simple. (They are equivalent to assuming perfect information.) The bounds can likely be significantly improved, allowing more pruning and faster search. We plan to implement improved bounds and evaluate the approach on a wider range of test problems. Our approach to determining the context of a decision node and merging duplicate OR nodes is also simple, and could potentially be improved, and it may also be possible to find more compact representations of an optimal strategy.

Acknowledgments This research is partially supported by NSF grants IIS-0953723 and IIS-1219114.

References

[1] C. P. de Campos and Q. Ji. Strategy selection in influence diagrams using imprecise probabilities. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-08)*, pages 121–128, 2008.

[2] R. Dechter and R. Mateescu. AND/OR search spaces for graphical models. *Artificial Intelligence*, 171(2-3):73–106, 2007.

[3] E. A. Hansen, D. S. Bernstein, and S. Zilberstein. Dynamic programming for partially observable stochastic games. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-04)*, pages 709–715, 2004.

[4] M. C. Horsch and D. Poole. An anytime algorithm for decision making under uncertainty. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, 1998.

[5] R. A. Howard and J. E. Matheson. Influence diagrams. In R. A. Howard and J. E. Matheson., editors, *The Principles and Applications of Decision Analysis*, pages 719–762, Menlo Park, CA, 1981.

[6] F. Jensen, F. V. Jensen, and S. L. Dittmer. From influence diagrams to junction trees. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence (UAI-94)*, pages 367–373, 1994.

[7] F. V. Jensen and T. D. Nielsen. *Bayesian Networks and Decision Graphs*, chapter 10, page 358. Springer Science+Business Media, LLC, New York, 2 edition, 2007.

[8] S. L. Lauritzen and D. Nilsson. Representing and solving decision problems with limited information. *Management Science*, 47(9):1235–1251, 2001.

[9] R. Marinescu. A new approach to influence diagrams evaluation. In *Proceedings of the 29th SGAI International Conference on Artificial Intelligence (AI-2009)*, pages 107–120, 2009.

[10] D. Mauá, C. de Campos, and M. Zaffalon. Solving limited memory influence diagrams. *Journal of Artificial Intelligence Research*, 44:97–140, 2012.

[11] D. D. Mauá and C. P. de Campos. Solving decision problems with limited information. In *Advances in Neural Information Processing Systems 24: Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS-11)*, 2011.

[12] R. Nair, M. Tambe, M. Yokoo, D. Pynadath, and S. Marsella. Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 705–711, 2003.

[13] T. D. Nielsen and F. V. Jensen. Well-defined decision scenarios. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 502–511, 1999.

[14] D. Nilsson and M. Hohle. Computing bounds on expected utilities for optimal policies based on limited information. Technical Report 94, Danish Informatics Network in the Agricultural Sciences, 2001.

[15] S. M. Olmsted. *On representing and solving decision problems*. PhD thesis, Stanford University, 1983.

[16] R. Qi and D. L. Poole. A new method for influence diagram evaluation. *Computational Intelligence*, 11:498–528, 1995.

[17] H. Raiffa and R. Schlaifer. *Applied Statistical Decision Theory*. MIT Press, Cambridge, 1961.

[18] S. Seuken and S. Zilberstein. Memory-bounded dynamic programming for DEC-POMDPs. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2009–2015, 2007.

[19] R. Shachter. Evaluating influence diagrams. *Operations Research*, 34:871–882, 1986.

[20] R. Shachter and M. Peot. Decision making using probabilistic inference methods. In *Proceedings of the 8th Conference on Uncertainty in Artificial Intelligence (UAI-92)*, pages 276–283, 1992.

[21] R. D. Shachter. Efficient value of information computation. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 594–601, 1999.

[22] J. E. Smith, S. Holtzman, and J. E. Matheson. Structuring conditional relationships in influence diagrams. *Operations Research*, 41:280–297, April 1993.

[23] C. Yuan and E. A. Hansen. Efficient computation of jointree bounds for systematic MAP search. In *Proceedings of 21st International Joint Conference on Artificial Intelligence (IJCAI-09)*, 2009.

[24] C. Yuan, X. Wu, and E. A. Hansen. Solving multistage influence diagrams using branch-and-bound search. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pages 691–700, 2010.

Constrained Bayesian Inference for Low Rank Multitask Learning

Oluwasanmi Koyejo
Electrical Engineering Dept.
University of Texas at Austin

Joydeep Ghosh
Electrical Engineering Dept.
University of Texas at Austin

Abstract

We present a novel approach for constrained Bayesian inference. Unlike current methods, our approach does not require convexity of the constraint set. We reduce the constrained variational inference to a parametric optimization over the feasible set of densities and propose a general recipe for such problems. We apply the proposed constrained Bayesian inference approach to multitask learning subject to rank constraints on the weight matrix. Further, constrained parameter estimation is applied to recover the sparse conditional independence structure encoded by prior precision matrices. Our approach is motivated by reverse inference for high dimensional functional neuroimaging, a domain where the high dimensionality and small number of examples requires the use of constraints to ensure meaningful and effective models. For this application, we propose a model that jointly learns a weight matrix and the prior inverse covariance structure between different tasks. We present experimental validation showing that the proposed approach outperforms strong baseline models in terms of predictive performance and structure recovery.

1 INTRODUCTION

The Bayesian paradigm has become one of the most important approaches for modeling uncertainty. Bayes' classic theorem provides a principle for updating prior beliefs with new information, and has become and important component of statistics and machine learning. Despite the elegance of Bayes theorem, some learning problems require model constraints that are difficult or inappropriate to enforce using standard prior distributions. Examples include linear inequality constraints (Gelfand et al., 1992) and margin constraints (Zhu et al., 2012).

Williams (1980) showed that the Bayesian posterior distribution can be derived as the solution of a constrained relative entropy minimization problem. Constrained Bayesian inference is proposed as an extension that combines Bayesian inference with other constraints determined by domain knowledge. Constrained relative entropy minimization can be solved via Fenchel duality theory (Altun & Smola, 2006), which requires convexity of the constraint set. In this paper, we propose an alternative approach that does not require the constraint set to be convex. We find that the optimization problem resulting from the proposed approach may be easier to solve than the equivalent Fenchel dual approach even when the constraint set is convex.

Rank constraints have proven to be effective for controlling model complexity (Candés & Recht, 2009). In addition to superior performance in many scenarios, the decompositions learned are often useful for explaining the structure of complex multivariate data. Low rank latent variable models have been applied to various domains including principal component analysis (Bishop, 1998), multitask learning (Stegle et al., 2011) and collaborative filtering (Salakhutdinov & Mnih, 2008). We propose a novel approach for low rank multitask learning via Bayesian inference subject to a nuclear norm constraint on the predictive weight matrix. The constrained inference is combined with parameter estimation for the prior precision of the matrix-variate Gaussian distribution. We enforce l_1 regularization constraints on the precision matrix to reveal its sparsity structure.

Our work is motivated by reverse inference for functional neuroimaging. Functional neuroimaging datasets typically consist of a relatively small number of correlated high dimensional brain images. Hence, capturing the inherent structural properties of the imaging data is critical for robust inference. Predictive modeling (also known as "brain reading" or "reverse inference") has become an increasingly popular approach for studying fMRI data (Pereira et al., 2009; Poldrack, 2011). Reverse inference involves the interpretation of the parameters of a model trained to decode the stimulus or task using the brain images as features. We show that the proposed approach is effective in this domain.

The contributions of this paper are as follows:

- We propose an a novel representation approach for constrained Bayesian inference. We prove that the optimizing density is a member of an exponential family. The presented results relax the necessary conditions on the constraint set such as convexity.
- We develop a novel constrained Bayesian model for rank constrained multitask learning and apply l_1 norm constrained parameter estimation to estimate the intertask conditional independence structure.
- The proposed multitask learning approach is applied to reverse inference for functional neuroimaging data. We show that the proposed approach results in superior accuracy as compared to strong baseline models.

As a minor contribution, our work appears to be the first application of constrained Bayesian inference to continuous valued variables that are not margin constraints. Constrained Bayesian inference is discussed in Section 2 and the proposed representation approach is introduced in Section 2.1. We discuss the proposed rank constrained multitask learning approach in Section 3. Related work is discussed in Section 4 and experimental results are presented in Section 5.

1.1 PRELIMINARIES

We denote vectors by lower case $\mathbf{x}$ and matrices by capital $\mathbf{X}$. Let $\mathbf{I}_D$ represent the $D \times D$ identity matrix. Given a matrix $\mathbf{A} \in R^{P\times Q}$, $\text{vec}(\mathbf{A}) \in R^{PQ}$ is the vector obtained by concatenating columns of $\mathbf{A}$. Given matrices $\mathbf{A} \in R^{P\times Q}$ and $\mathbf{B} \in R^{P'\times Q'}$, the Kronecker product of $\mathbf{A}$ and $\mathbf{B}$ is denoted as $\mathbf{A} \otimes \mathbf{B} \in R^{PP'\times QQ'}$. We use $\|\cdot\|_p$ to denote the vector L_p norm with $\|\mathbf{x}\|_p = (\sum_i x_i^p)^{\frac{1}{p}}$, and use $|\!|\!|\cdot|\!|\!|$ to denote spectral (matrix) norms i.e. $|\!|\!|\mathbf{X}|\!|\!|_p$ is the L_p norm of the singular values of $\mathbf{X}$.

Let $\mathcal{X}$ be a Banach space and let $\mathcal{X}^*$ be the *dual space* of $\mathcal{X}$. The *Legendre-Fenchel* transformation (or convex conjugate) of a function $f : \mathcal{X} \mapsto [-\infty, +\infty]$ is given by $f^* : \mathcal{X}^* \mapsto [-\infty, +\infty]$ as $f^*(x^*) = \sup_{x\in\mathcal{X}}\{\langle x, x^*\rangle - f(x)\}$. where $\langle x, x^*\rangle$ denotes the dual pairing. See Borwein & Zhu (2005) for further details on Fenchel duality, particularly as applied to variational optimization.

Let E be the *expectation operator* with $\text{E}_p[f(z)] = \int_z p(z)f(z)dz$. The *Kullback-Leibler divergence* between densities q and p is given by $\text{KL}(q(z)\|p(z)) = \text{E}_{q(z)}[\log q(z) - \log p(z)]$. The *delta function* as a generalized function that satisfies $\int_Z f(z)\delta_a\{dz\} = f(a)$, where f is absolutely continuous with respect to dz, and $a \in Z$. Following from the definition, the expectation with respect to the delta function satisfies $\text{E}_{\delta_a}[f] = f(a)$, and given the density p, we have that $\text{E}_p[\delta_a] = p(a)$. Further, it can be shown (Williams, 1980) that $\text{KL}(\delta_a\|p) = -\log p(a)$.

An *exponential family* is a class of probability distributions whose density functions take the form:

$$p(\mathbf{x}|\boldsymbol{\theta}) = h(\mathbf{x})e^{\langle\boldsymbol{\eta}(\boldsymbol{\theta}),\mathbf{t}(\mathbf{x})\rangle - G(\boldsymbol{\theta})},$$

where $\boldsymbol{\eta}(\boldsymbol{\theta})$ is known as the natural parameter vector, $\mathbf{t}(\mathbf{x})$ is the vector of natural statistics, $G(\boldsymbol{\theta})$ is the log partition function and $h(\mathbf{x})$ is known as the base measure. The exponential family is in *canonical* form if $\boldsymbol{\eta}(\boldsymbol{\theta}) = \boldsymbol{\theta}$. Further details on exponential family distributions may be found in (Brown, 1986).

Let $\mathbf{x} \in R^D$ be drawn from a *multivariate Gaussian* distribution. The density is given as:

$$\mathcal{N}(\mathbf{m}, \boldsymbol{\Sigma}) = \frac{\exp\left(-\frac{1}{2}\text{tr}\left((\mathbf{x}-\mathbf{m})^\top\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{m})\right)\right)}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{P/2}},$$

where $\mathbf{m} \in R^D$ is the mean vector and $\boldsymbol{\Sigma} \in R^{D\times D}$ is the covariance matrix. $|\cdot|$ denotes the matrix determinant and $\text{tr}(\cdot)$ denotes the matrix trace. Let $\mathbf{X} \in R^{D\times K}$ be drawn from a *matrix-variate Gaussian* distribution represented as $\mathcal{MN}(\mathbf{M}, \boldsymbol{\Sigma}_R, \boldsymbol{\Sigma}_C)$ where $\mathbf{M} \in R^{D\times K}$ is the mean matrix, $\boldsymbol{\Sigma}_R \in R^{D\times D}$ is the row covariance matrix and $\boldsymbol{\Sigma}_C \in R^{K\times K}$ is the column covariance matrix. The density is given by:

$$\frac{\exp\left(-\frac{1}{2}\text{tr}\left(\boldsymbol{\Sigma}_C^{-1}(\mathbf{X}-\mathbf{M})^\top\boldsymbol{\Sigma}_R^{-1}(\mathbf{X}-\mathbf{M})\right)\right)}{(2\pi)^{DL/2}|\boldsymbol{\Sigma}_R|^{D/2}|\boldsymbol{\Sigma}_C|^{K/2}}.$$

2 CONSTRAINED BAYESIAN INFERENCE

Constrained relative entropy inference follows from the *principle of minimum discrimination information* (Kullback, 1959); a conceptual framework for updating a distribution given constraints. It defines a procedure for updating the distribution as one that satisfies the constraints and is closest to a predefined prior distribution in terms of relative entropy. Bayesian inference is recovered from the constrained relative entropy framework when the data constraints correspond to knowledge of the value of y with certainty (Williams, 1980). Given the observation $\tilde{y} \sim P_y$, this knowledge is encoded using the constraint $\text{E}_q[\delta_{\tilde{y}}] = 1$ that must be satisfied by the updated distribution q. The resulting constrained relative entropy minimization problem is given by:

$$\min_{q\in\mathcal{P}}\left[\text{KL}(q(z,y)\|p(z,y))\;\text{s.t.}\;\text{E}_q[\delta_{\tilde{y}}] = 1\right]. \qquad (1)$$

It is clear that any distribution q that optimizes (1) must satisfy the equivalent conditional distribution constraint

$q(y) = \int_Z q(z, y) = \delta_{\tilde{y}}$, so we will focus on estimating the portion of the distribution that remains unknown, which, from the basic rules of probability, is the conditional distribution $q(z|y)$. Thus, it will be useful to express the joint relative entropy in a form that separates the latent variables from the observations by expressing $\mathrm{KL}(q(z, y)\|p(z, y))$ as:

$$\mathrm{E}_{q(y)}\left[\mathrm{KL}(q(z|y)\|p(z|y))\right] + \mathrm{KL}(q(y)\|p(y)).$$

Enforcing the constraint $q(y) = \delta_{\tilde{y}}$, we recover that $\mathrm{KL}(q(x|y)q(y)\|p(x|y)p(y))$ is given by:

$$\mathrm{KL}(q(x|y = \tilde{y})\|p(x|y = \tilde{y})) - \log(p(\tilde{y})).$$

The second term $\log(p(\tilde{y}))$ is the log evidence, and is fixed independent of the first term. The first term is minimized when $q(x|y = \tilde{y}) = p(x|y = \tilde{y})$, recovering the Bayesian posterior distribution[1]. Thus, the solution of the relative entropy minimization problem (1) takes the form of the generalized density $q_*(x, y) = p(x|y = \tilde{y})\delta_{\tilde{y}}$.

For the rest of the discussion, we focus on q as a density with respect to Z (ignoring the implicit conditioning). It is instructive to expand the terms of the loss function. The relative entropy expands as follows:

$$\begin{aligned} &\mathrm{KL}(q(z)\|p(z|y)) - \log p(y) && \text{(2a)}\\ &= \mathrm{E}_q\left[\log q(z) - \log p(z|y) - \log p(y)\right] && \text{(2b)}\\ &= \mathrm{E}_q\left[\log q(z) - \log p(z) - \log p(y|z)\right] && \text{(2c)}\\ &= \mathrm{KL}(q(z)\|p(z)) - \mathrm{E}_q\left[\log p(y|z)\right]. && \text{(2d)} \end{aligned}$$

where (2b) and (2d) follow directly by expansion of the KL divergence, and (2c) follows from the rules of conditional probability as $p(z|y)p(y) = p(z, y) = p(y|z)p(z)$. The result of (2c) also recovers the identity discovered by Zellner (1988), who showed that the Bayesian posterior density is given by:

$$p(z|y) = \arg\min_{q\in\mathcal{P}} \mathrm{KL}(q(z)\|p(z)) - \mathrm{E}_q\left[\log p(y|z)\right]. \quad (3)$$

Constrained Bayesian inference defines a procedure for enforcing constraints on latent variables in addition to the constraints on the observation variables. Let $\boldsymbol{\beta}$ represent *feature functions* that map Z to a feature space with components $\boldsymbol{\beta}(z) = \{\beta_j(z)\}$ and let C denote a constraint set of interest. We consider information encoded as expectation constraints in this paper. The *constrained Bayesian inference* procedure is defined by the following equivalent optimization problems:

$$\min_{q\in\mathcal{P},\ \mathrm{E}_q[\boldsymbol{\beta}(z)]\in\mathsf{C}} \left[\mathrm{KL}(q(z)\|p(z|y))\right] \quad (4a)$$

$$\min_{q\in\mathcal{P},\mathrm{E}_q[\boldsymbol{\beta}(z)]\in\mathsf{C}} \left[\mathrm{KL}(q(z)\|p(z)) - \mathrm{E}_q\left[\log p(y|z)\right]\right] \quad (4b)$$

[1]Recall Bayes rule: $p(z|y) = p(y|z)p(z)/p(y)$

It is clear from (4a) that constrained Bayesian inference corresponds to an information projection of the Bayesian posterior distribution to the set to distributions q that satisfy the constraints $\mathrm{E}_q\left[\boldsymbol{\beta}(z)\right] \in \mathsf{C}$. Following Zellner, we call q_* the *postdata* distribution to distinguish it from the unconstrained Bayesian posterior distribution.

We now consider probabilistic inference via constrained relative entropy minimization. Altun & Smola (2006) studied norm ball constraints given by $\|\mathrm{E}_q\left[\boldsymbol{\beta}(z)\right)] - \mathbf{b}\|_{\mathcal{B}} \leq \epsilon$ where $\|\cdot\|_{\mathcal{B}}$ is the norm ball on a the Banach space $\mathcal{B}$ centered at $\mathbf{b} \in \mathcal{B}$, and $\epsilon \geq 0$ is the width. The solution was found by an elegant application of Fenchel duality for variational optimization (Borwein & Zhu, 2005). The following Lemma characterizes relative entropy minimization subject to norm ball constraints.

Lemma 1 (Altun & Smola (2006))**.**

$$\min_{q\in\mathcal{P}} \mathrm{KL}(q(z)\|p(z)) \text{ s.t. } \|\mathrm{E}_q\left[\boldsymbol{\beta}(z)\right] - \mathbf{b}\|_{\mathcal{B}} \leq \epsilon \quad (5)$$

$$= \max_{\boldsymbol{\lambda}} \langle\boldsymbol{\lambda}, \mathbf{b}\rangle - \log\int_Z p(z)e^{\langle\boldsymbol{\lambda},\boldsymbol{\beta}(z)\rangle}dz - \epsilon\|\boldsymbol{\lambda}\|_{\mathcal{B}^*} + e^{-1} \quad (6)$$

and the unique solution is given by $q_*(z) = p(z)e^{\langle\boldsymbol{\lambda}_*,\boldsymbol{\beta}(z)\rangle - G(\boldsymbol{\lambda}_*)}$ *where* $\boldsymbol{\lambda}_*$ *is the solution of the dual optimization* (6) *and* $G(\boldsymbol{\lambda}_*)$ *ensures normalization.*

There may be several equivalent representations for a given density $q \in \mathcal{P}$. However, Lemma 1 shows that the density that minimizes relative entropy subject to norm ball constraints, if it exists, has a canonical representation a member of the exponential family with base measure p, natural statistics $\boldsymbol{\beta}(z)$ and parameters $\boldsymbol{\lambda}_*$. The conditions for Lemma 1 include constraint qualification, which requires the existence of densities that satisfy the set of constraints, and a finite cost (6) at the solution $\boldsymbol{\lambda}_*$. More details are given in Altun & Smola (2006) and Chapter 4 of Borwein & Zhu (2005).

2.1 A REPRESENTATION APPROACH

The dual solution presented in Lemma 1 requires convexity of the constraint set C. Further, solving the resulting dual optimization (6) requires the evaluation of the log partition function which is often challenging. We present an alternative representation approach that separates the problem into two parts. First we find the parametric family of the optimizing postdata density, then we directly optimize over that parametric family. Unlike the dual approach, the proposed representation approach does not require convexity of the constraint set.

For the rest of this paper, we will assume that the set of solutions q_* of the constrained Bayesian optimization (4) is not empty so the optimization problem is well defined. This implies the existence of at least one density $q \in \mathcal{P}$ that satisfies the constraints $\mathrm{E}_q\left[\boldsymbol{\beta}(z)\right] \in \mathsf{C}$. This also implies that the solution of the variational optimization problem is achieved at a density q_*. Further, we assume that for each solution q_*, the expectation $\mathrm{E}_{q_*}\left[\boldsymbol{\beta}(z))\right] = \mathbf{a}_*$ is bounded to avoid the degenerate problem of unbounded constraints. Finally, we assume that $\mathsf{C} \subset \mathcal{B}$ is a closed subset of the Banach space $\mathcal{B}$. This assumption is mostly for convenience and clarity and can easily be relaxed.

Let $\mathcal{E}_{\mathbf{c}} = \{q \in \mathcal{P} \,|\, \mathrm{E}_q\left[\boldsymbol{\beta}(z)\right] = \mathbf{c}\}$ denote the constraint set subject to equality constraints. The constrained Bayes optimization problem (4b) can be written as:

$$\min_{\mathbf{c}\in\mathsf{C}}\left[\min_{q\in\mathcal{E}_{\mathbf{c}}} \mathrm{KL}(q(z)\|p(z|y))\right], \tag{7}$$

which requires the solution of an inner optimization:

$$q_{\mathbf{c}} = \arg\min_{q\in\mathcal{E}_{\mathbf{c}}} \mathrm{KL}(q(z)\|p(z|y)). \tag{8}$$

Let $\mathsf{A} \subset \mathsf{C}$ represent the set of points $\mathbf{c} \in \mathsf{C}$ where $\mathbf{c}$ is bounded, and the optimization problem of (4b) is finite and attained. Assuming the existence of at least one solution q_*, it follows that the set A is not empty. We associate a density function $q_{\mathbf{c}}$ to every element $\mathbf{c} \in \mathsf{A}$. We define a *feasible set* of solutions characterized by the set of densities $\mathcal{F} = \{q_{\mathbf{c}}(z) \,|\, \mathbf{c} \in \mathsf{A}\}$. The following proposition is a direct consequence of Lemma 1 and is stated without proof.

Proposition 2. *For any* $\mathbf{c} \in \mathsf{A}$*, the unique minimizer of* (8) *is given by:* $q_{\mathbf{c}}(z) = p(z|y)e^{\langle\boldsymbol{\lambda}_{\mathbf{c}},\boldsymbol{\beta}(z)\rangle - G(\boldsymbol{\lambda}_{\mathbf{c}})}$ *where* $\boldsymbol{\lambda}_{\mathbf{c}}$ *is the solution of the dual optimization* (6) *with* $\epsilon = 0$ *and* $G(\boldsymbol{\lambda}_{\mathbf{c}})$ *ensures normalization.*

We may now state the main result.

Theorem 3. *Let* $\mathcal{F} = \{q_{\mathbf{c}} \,|\, \mathbf{c} \in \mathsf{A}\}$ *denote the feasible set of* (8)*. The postdata density given by the minimizer of* (4a) *is the solution of:*

$$q_* = \arg\min_{q\in\mathcal{F}} \mathrm{KL}(q(z)\|p(z|y))$$

and the solution is given by $q_* = q_{\mathbf{a}_*}$ *for the optimal* $\mathbf{a}_* \in \mathsf{A}$ *with* $q_*(z) = p(z|y)e^{\langle\boldsymbol{\lambda}_{\mathbf{a}_*},\boldsymbol{\beta}(z)\rangle - G(\boldsymbol{\lambda}_{\mathbf{a}_*})}$ *where* $\boldsymbol{\lambda}_{\mathbf{a}_*}$ *is the solution of the dual optimization* (6) *with the constraint set* $\mathsf{C}' = \{\mathrm{E}_q\left[\boldsymbol{\beta}(z)\right] = \mathbf{a}_*\}$ *and* $G(\boldsymbol{\lambda}_{\mathbf{a}_*})$ *ensures normalization. The solution is unique if* A *is convex.*

Sketch of proof: First we prove that $q_* \in \mathcal{F}$ by contradiction. Suppose $q_* \notin \mathcal{F}$, then $q_* = q_{\mathbf{v}}$ for $\mathbf{v} \notin \mathsf{A}$. This is a contradiction by definition of A. The first claim follows directly. The parametric form of the solution follows from Proposition 2 and the uniqueness of the solution for convex A follows from the strict convexity of the relative entropy.

Applied directly, Theorem 3 requires the solution of the equality constrained variational inference (8) in the inner loop. The key insight from Proposition 2 is that the solution of (8) fully specifies the parametric form of the density. In other words, all the members of the set $\mathcal{F} = \{q_{\mathbf{c}} \,|\, \mathbf{c} \in \mathsf{A}\}$ have the same parametric form f where $q_{\mathbf{c}} = f_{\boldsymbol{\theta}_{\mathbf{c}}}(z)$ is determined by the choice of $\mathbf{c}$. By definition of the exponential family, all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ where $\boldsymbol{\Theta}$ is the constraint set of the parametric distribution family containing $f_{\boldsymbol{\theta}}$.

Corollary 4. *The postdata density is the minimizer of* (4a) *and is given by* $q_* = f_{\boldsymbol{\theta}_*}$ *where* $\boldsymbol{\theta}_*$ *is the solution of:*

$$\boldsymbol{\theta}_* = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}}\left[\begin{array}{l}\mathrm{KL}(f_{\boldsymbol{\theta}}(z)\|p(z|y))\\ \text{s.t. } \mathrm{E}_{f_{\boldsymbol{\theta}}}\left[\boldsymbol{\beta}(z)\right] \in \mathsf{C}\end{array}\right].$$

Sketch of proof: Let the exponential family $\mathcal{G} = \{f_{\boldsymbol{\theta}} \,|\, \forall\, \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$. Clearly $\mathcal{F} \subseteq \mathcal{G}$ by definition. Suppose $f_{\boldsymbol{\theta}_*} \notin \mathcal{F}$, feasibility implies that $f_{\boldsymbol{\theta}_*}$ satisfies the constraints. Thus $\exists\ \mathbf{v}$ such that $\mathrm{E}_{f_{\boldsymbol{\theta}_*}}\left[\boldsymbol{\beta}(z)\right] = \mathbf{v}$ and $\mathbf{v} \notin \mathsf{A}$. This is a contradiction by definition of A. Thus $f_{\boldsymbol{\theta}_*} = q_* \in \mathcal{F} \subset \mathcal{G}$ and the proof follows from Theorem 3.

Our approach suggests the following *recipe* for constrained Bayesian inference. First, Proposition 2 is applied to specify the parametric form of q_*, then Corollary 4 is applied to convert the variational problem into a parametric optimization problem.

3 RANK CONSTRAINED MULTITASK LEARNING

Let $n = 1 \ldots N$ denote the number of training examples and $k = 1 \ldots K$ denote each task so that the output is given by $y_{n,k} \in R$. Given a D dimensional feature vector $\mathbf{x}_n \in R^D$ and a weight vector $\mathbf{w}_k \in R^D$, each output is generated as:

$$y_{n,k} = \mathbf{w}_k^\top \mathbf{x}_n + \epsilon$$

where $\epsilon \sim \mathcal{N}\left(0, \sigma^2\right)$. The outputs may be collected into a output matrix $\mathbf{Y} \in R^{N\times K}$ and the features may be collected into a feature matrix $\mathbf{X} \in R^{D\times K}$ with $\mathbf{X}(n) = \mathbf{x}_n^\top$. The latent matrix $\mathbf{W}$ is drawn from a zero mean matrix-variate Gaussian distribution $\mathbf{W} \sim \mathcal{MN}\left(\mathbf{0}, \mathbf{R}, \mathbf{C}\right)$ with row covariance $\mathbf{R} \in R^{D\times D}$ and column covariance matrix $\mathbf{C} \in R^{K\times K}$. Fig. 1 illustrates the combined generative model. Without loss of generality, we assume that the output matrix is normalized to zero mean over the columns so we do not include a bias term. The model parameters are given by $\boldsymbol{\Theta} = \{\mathbf{R}, \mathbf{C}, \sigma^2\}$.

The unconstrained posterior distribution can be computed in closed form (Bishop, 2006). Let $\mathbf{w} = \mathrm{vec}(\mathbf{W})$ and $\mathbf{y} = \mathrm{vec}(\mathbf{Y})$, then $p(\mathbf{w}|\mathbf{y}) = \mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$ where:

$$\boldsymbol{\mu} = \frac{1}{\sigma^2}\boldsymbol{\Sigma}(\mathbf{I}_K \otimes \mathbf{X}^\top)\mathbf{y} \tag{9}$$

$$\boldsymbol{\Sigma}^{-1} = (\mathbf{C}^{-1} \otimes \mathbf{R}^{-1}) + \frac{1}{\sigma^2}(\mathbf{I}_K \otimes \mathbf{X}^\top\mathbf{X}) \tag{10}$$

with $\boldsymbol{\mu} \in R^{DK}$ and $\boldsymbol{\Sigma} \in R^{DK\times DK}$.

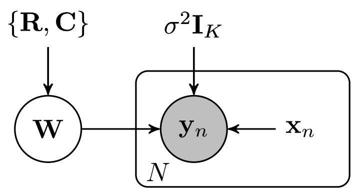


Figure 1: Generative Model for Multitask Learning

3.1 CONSTRAINED INFERENCE

We seek to enforce a rank constraint via the constraint set $\mathsf{B} = \{\mathbf{B} \,|\, \text{rank}(\mathbf{B}) \leq R\}$. We apply the recipe discussed in Section 2.1. First we must define the *parametric form* of the postdata distribution by solving (8) for a fixed $\mathbf{b} \in \mathsf{B}$. We find that $q_{\mathbf{b}}(\mathbf{w})$ is Gaussian distributed with density $\mathcal{N}(\mathbf{m}, \mathbf{S})$ and $\mathbf{m} = \mathbf{b}$. Following the arguments of Corollary 4, the postdata distribution is found by minimizing the KL divergence between the Gaussian distribution $\mathcal{N}(\mathbf{m}, \mathbf{S})$ and the Bayesian posterior distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. This is given by:

$$\min_{\mathbf{m}\in\mathsf{B},\mathbf{S}} \text{tr}\big(\boldsymbol{\Sigma}^{-1}\mathbf{S}\big) + (\boldsymbol{\mu} - \mathbf{m})^\top\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \mathbf{m}) - \log|\mathbf{S}| + \log|\boldsymbol{\Sigma}| \qquad (11)$$

where $\mathbf{m} = \text{vec}(\mathbf{M})$. The optimization decouples between the mean term $\mathbf{m}$ and the covariance term $\mathbf{S}$. The minimum in terms of the covariance is achieved for $\mathbf{S} = \boldsymbol{\Sigma}$ and the mean optimization is given by the solution of a rank constrained quadratic optimization. We note that the Gaussian form of the constrained postdata density was not assumed *a-priori*, but was found as the solution to the constrained inference.

The solution of (11) requires computation and storage of the posterior covariance $\boldsymbol{\Sigma}$. This may become computationally infeasible for high dimensional data. In such situations, it may be more computationally efficient to estimate the postdata mean matrix using the form of (4b). Ignoring terms independent of the mean, this results in the optimization problem:

$$\min_{\mathbf{M}\in\mathsf{B}} \frac{1}{\sigma^2}|||\mathbf{Y} - \mathbf{X}\mathbf{M}|||_2^2 + \text{tr}\big(\mathbf{M}^\top\mathbf{R}^{-1}\mathbf{M}\mathbf{C}^{-1}\big) \qquad (12)$$

Nuclear norm constraint: The rank constraint is nonconvex and is challenging to optimize directly. To simplify the optimization, we replace the rank constraint with a nuclear norm constraint $\mathsf{D} = \{\mathbf{D} \,|\, |||\mathbf{D}|||_1 \leq C\}$. The nuclear norm is computed as sum of the singular values of the matrix i.e. $|||\mathbf{D}|||_1 = \sum \sigma_i(\mathbf{D})$ where $\sigma_i(\mathbf{D})$ is the i^{th} singular value of the matrix $\mathbf{D}$. The nuclear norm is known to encourage low rank solutions Candés & Recht (2009). The resulting postdata mean inference retains the same form with the new constraint set. Replacing the constraint set with a regularization function, we find that the postdata mean optimization can be rewritten as:

$$\min_{\mathbf{M}} \frac{1}{\sigma^2}|||\mathbf{Y} - \mathbf{X}\mathbf{M}|||_2^2 + \text{tr}\big(\mathbf{M}^\top\mathbf{R}^{-1}\mathbf{M}\mathbf{C}^{-1}\big) + |||\mathbf{M}|||_1 \qquad (13)$$

We note that there is no need for a regularization parameter if we learn the hyperparameters $\boldsymbol{\Theta} = \{\mathbf{R}, \mathbf{C}, \sigma^2\}$, as the optimization only depends on the relative scale of the three terms.

Kronecker Covariance constraint: Unlike the prior covariance, the posterior covariance matrix does not decompose into Kronecker form. Hence, the size of the posterior covariance may be of computational concern. We propose a Kronecker factorization constraint structure for the posterior covariance matrix. Following Theorem 3, we find that the postdata distribution retains its Gaussian form. Let $\mathbf{S} = \mathbf{H} \otimes \mathbf{G}$ where $\mathbf{G} \in R^{D\times D}$ is constrained row covariance matrix and $\mathbf{H} \in R^{K\times K}$ is the constrained column covariance matrix. Employing the cost function (4b) and ignoring terms independent of the postdata covariance, we compute:

$$\min_{\mathbf{G},\mathbf{H}} \frac{1}{\sigma^2}\text{tr}\big(\mathbf{X}^\top\mathbf{X}\mathbf{G}\big)\,\text{tr}(\mathbf{H}) + \text{tr}\big(\mathbf{R}^{-1}\mathbf{G}\big)\,\text{tr}\big(\mathbf{C}^{-1}\mathbf{H}\big) - K\log|\mathbf{G}| - D\log|\mathbf{H}|$$

This can be solved using an alternating optimization approach:

$$\mathbf{G}^{-1} = \frac{1}{K}\left(\frac{\text{tr}(\mathbf{H})}{\sigma^2}\mathbf{X}^\top\mathbf{X} + \text{tr}\big(\mathbf{C}^{-1}\mathbf{H}\big)\,\mathbf{R}^{-1}\right) \qquad (14)$$

$$\mathbf{H}^{-1} = \frac{1}{D}\left(\frac{\text{tr}\big(\mathbf{X}^\top\mathbf{X}\mathbf{G}\big)}{\sigma^2}\mathbf{I}_K + \text{tr}\big(\mathbf{R}^{-1}\mathbf{G}\big)\,\mathbf{C}^{-1}\right) \qquad (15)$$

The result of constrained inference is the postdata distribution $q_*(\mathbf{W}|\mathbf{Y}) = \mathcal{MN}(\mathbf{M}, \mathbf{G}, \mathbf{H})$.

3.2 PARAMETER ESTIMATION

In addition to low rank constraints on the weight matrix, we are interested in learning the prior conditional independence structure between the features and between the tasks. This is achieved by placing Laplacian priors (Friedman et al., 2008; Stegle et al., 2011) on the row and column prior precision matrices:

$$p(\mathbf{R}^{-1}) \propto \exp(-\lambda_r\|\mathbf{R}^{-1}\|_1)[\mathbf{R}^{-1} \succ 0],$$
$$p(\mathbf{C}^{-1}) \propto \exp(-\lambda_c\|\mathbf{C}^{-1}\|_1)[\mathbf{C}^{-1} \succ 0],$$

where the l_1 norm is given by $\|\mathbf{R}\|_1 = \sum_{i,j}|r_{ij}|$. Ignoring terms independent of the precision matrices, the loss function is given by:

$$\min_{\mathbf{R}^{-1},\mathbf{C}^{-1}} \text{tr}\big(\mathbf{R}^{-1}\mathbf{G}\big)\,\text{tr}\big(\mathbf{C}^{-1}\mathbf{H}\big) + \text{tr}\big(\mathbf{W}^\top\mathbf{R}^{-1}\mathbf{W}\mathbf{C}^{-1}\big) - K\log|\mathbf{R}| - D\log|\mathbf{C}| + \lambda_r\|\mathbf{R}^{-1}\|_1 + \lambda_c\|\mathbf{C}^{-1}\|_1 \qquad (16)$$

Algorithm 1 Constrained Inference and Parameter Estimation for Multitask Learning

```
Initialize $\mathbf{G}$, $\mathbf{H}$, $\mathbf{\Theta} = \{\mathbf{R}, \mathbf{C}, \sigma^2\}$
repeat
  Update $\mathbf{M}|\mathbf{\Theta}$ by solving (13) (equiv. (11) or (12))
  repeat
    Update $\mathbf{G}|\mathbf{H}, \mathbf{\Theta}$ using (14)
    Update $\mathbf{H}|\mathbf{G}, \mathbf{\Theta}$ using (15)
  until converged
  repeat
    Update $\mathbf{R}|\mathbf{C}, \mathbf{G}, \mathbf{H}, \lambda_r$ by optimizing (16)
    Update $\mathbf{C}|\mathbf{R}, \mathbf{G}, \mathbf{H}, \lambda_c$ by optimizing (16)
  until converged
  Update $\sigma^2|\mathbf{M}, \mathbf{G}, \mathbf{H}$ using (17)
until converged
Return $\mathbf{M}$, $\mathbf{G}$, $\mathbf{H}$, $\mathbf{\Theta}$
```

We apply an alternating optimization approach, alternating between solving for $\mathbf{R}^{-1}$ and $\mathbf{C}^{-1}$. Each of these sub-optimization problems can be solved using *glasso* (Friedman et al., 2008)

Noise variance update: We also may also update the output noise variance. Ignoring terms independent of the noise variance, the optimization is given by minimizing (with respect to σ^2):

$$ND \log \sigma^2 + \frac{1}{\sigma^2}\left[\|\|\mathbf{Y} - \mathbf{X}\mathbf{W}\|\|_2^2 + \mathrm{tr}\big(\mathbf{X}^\top \mathbf{X}\mathbf{G}\big)\,\mathrm{tr}(\mathbf{H})\right]$$

This can be solved in closed form. The solution is given by:

$$\sigma^2 = \frac{1}{ND}\left[\|\|\mathbf{Y} - \mathbf{X}\mathbf{W}\|\|_2^2 + \mathrm{tr}\big(\mathbf{X}^\top \mathbf{X}\mathbf{G}\big)\,\mathrm{tr}(\mathbf{H})\right] \quad (17)$$

3.3 ALGORITHM

Our goal is to minimize the cost function (4). We solve this by alternating between constrained inference and parameter estimation. Constrained inference involves estimation of the postdata distribution $q(\mathbf{W}|\mathbf{Y})$ subject to rank (nuclear norm) and Kronecker covariance constraints, and constrained parameter estimation involves the estimation of updated parameters $\mathbf{\Theta}$. The proposed algorithm is summarized in Algorithm 1.

4 RELATED WORK

Examples of constrained Bayesian inference in the literature include maximum entropy discrimination (Jaakkola et al., 1999) and posterior regularization Ganchev et al. (2010). Ganchev et al. (2010) applied constrained Bayesian inference techniques to statistical word alignment, multiview learning, dependency parsing and part of speech induction. More recently, researchers have applied constrained Bayesian inference for combining complicated nonparametric topic models with support vector machine inspired large margin constraints for document classification (Zhu et al., 2009), multitask classification (Zhu et al., 2011) and link prediction (Zhu, 2012).

Constrained Bayesian inference is closely related to techniques for approximate variational Bayesian inference (Bishop, 2006), used to approximate intractable Bayesian posterior densities. The approximation typically takes the form of factorization assumptions between subsets of the latent variables. The result is often much easier to solve. Although approximate variational inference also requires solving a constrained version of (3), the motivations and results are quite different than in constrained Bayesian inference methods. In particular, the estimated constrained Bayes distributions may not factorize over subsets of the latent variables.

Partial Least squares (PLS) (Abdi, 2010) is a popular approach for low rank multiple regression. PLS estimates low rank factors that best matches the cross correlation between the features and the response and is known to be especially effective when the feature matrix has co-linear rows and when the features are very high dimensional. Argyriou et al. (2007) and Yuan et al. (2007) proposed models for multitask learning using a regularizer that penalizes the nuclear norm of the weight matrix. This constraint often results in a weight matrix of low rank. Rai & Daumé III (2010) proposed a nonparametric Bayesian model for multitask learning using the direct low rank factor representation. The proposed approach is able to estimate the number of factors using the Indian buffet process prior.

(Zhang & Schneider, 2010; Allen & Tibshirani, 2012) studied covariance estimation for the matrix-variate Gaussian distribution subject to l_1 constraints on the precision when the observed data was generated directly from a matrix-variate Gaussian distribution. Stegle et al. (2011) extended the work to the case where the matrix-variate Gaussian distribution is used a the prior, coupled with additive noise. They showed that capturing the additive noise structure can make a significant impact on the quality of the recovered precision matrix. They noted the in difficulty of inference in the model and proposed a heuristic using only the posterior mean and ignoring the posterior covariance. Following our development, the heuristic inference approach of Stegle et al. (2011) can now be explained as a constrained Bayesian inference subject to the constraint the the posterior covariance vanishes. We compare the performance of this heuristic with the Kronecker constrained inference approach on simulated and real data experiments, showing the utility of the richer posterior covariance structure.

5 EXPERIMENTS

We present experimental results comparing the proposed rank constrained variational approach to other matrix-variate learning models in the literature. We compared the models in terms of regression accuracy and in terms of structure recovery for the underlying precision matrices. The compared models are as follows:

- Graphical Lasso (GLasso) (Friedman et al., 2008) estimates a sparse precision matrix to match the sample covariance of the response matrix. GLasso was used a the baseline for inter-task structure recovery.
- Multiple regularized ridge regression (Ridge) was used a the baseline for the regression accuracy. Ridge does not estimate the precision matrix.
- Partial least squares (PLS) (Abdi, 2010) estimates low rank factors that best matches the cross correlation between the features and the response. The resulting weight vector can be used for prediction. PLS does not estimate the precision matrix.
- Nuclear norm regularized linear regression (Nuc. Norm) (Yuan et al., 2007) estimates the regression matrix that best predicts the target response subject to a nuclear norm constraint. The resulting weight matrix is often of low rank. Nuc. Norm does not estimate the precision matrix.
- We implemented the matrix variate regression and sparse precision matrix estimation procedure of (Stegle et al., 2011) (MVG). Our approach fixed the feature matrix instead of estimating it from data. As noted in Section 4, Stegle et al. (2011) used a heuristic procedure with a degenerate posterior covariance for the model inference. There is no low rank constraint applied to the model.
- We implemented a *corrected* matrix variate regression and sparse precision matrix estimation procedure using the Kronecker product posterior covariance constraint proposed in Section 3.1 ($\text{MVG}_{\text{corr.}}$). There is no low rank constraint applied to the model.
- We implemented the proposed nuclear norm constrained matrix variate regression and sparse precision matrix estimation using the constrained Bayesian inference approach regression (MVG_{rank}). This combines the nuclear norm constrained inference for the low rank weight matrix with l_1 constrained precision matrix learning.

We optimized the proposed MVG_{rank} model by using the approach outlined in Algorithm 1. A similar procedure without the nuclear norm constraint was used to optimize the $\text{MVG}_{\text{corr.}}$ and MVG models. Nuc. Norm was optimized using a special case of MVG_{rank} without any of the covariance matrices. GLasso, Ridge and PLS were optimized using implementations from the *scikit-learn* python package (Pedregosa et al., 2011).

5.1 SIMULATED DATA

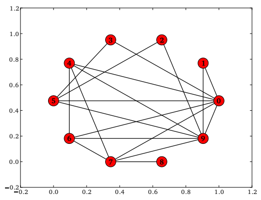
Constrained inference is most useful when data is scarce. We are most interested in high dimensional multiple regression where there are more dimensions than samples. In such scenarios, the model constraints can be critical for effective regression and parameter estimation. We performed experiments using simulated data that matches the characteristics of functional neuroimaging data. We fixed the row precision matrix and tested the models ability of estimate the structure of the column precision matrix and the predictive accuracy of the model.

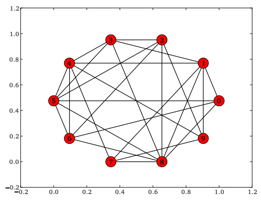
We generated a random row precision matrix by generating using the approach outlined in Example 1 of Li & Toh (2010). We first generated a sparse matrix $\mathbf{U}$ with non zero entries equal to $+1$ or -1, then set $\mathbf{C}^{-1} = \mathbf{U}\mathbf{U}^{\top}$. Finally, we added a diagonal term to ensure $\mathbf{C}^{-1}$ is positive definite. The resulting column precision matrix had a sparsity of 20%. The column precision was generated as the normalized Laplacian matrix (Smola & Kondor, 2003) of a chain graph with a adjacency matrix set as $\mathbf{A}_{i,j} = 1$ if $j = \{i, i+1, i-1\}$ and zero otherwise.

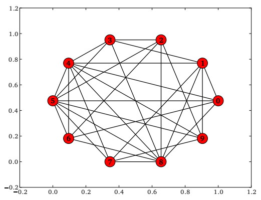
We generated a low rank weight matrix using the factor model as $\mathbf{W} = \mathbf{A}\mathbf{B}^{\top}$. The columns of $\mathbf{A}$ were generated from the zero mean multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{R})$ and the columns of $\mathbf{B}$ were generated from the zero mean multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{C})$. We also generated random high dimensional feature matrices $\mathbf{X} \in R^{N \times D}$ with $x_{i,j} \sim \mathcal{N}(0,1)$. Finally the response matrix was generated as $\mathbf{Y} = \mathbf{X}\mathbf{W} + \mathbf{N}$ where $\mathbf{N}$ represents independent additive noise with each entry $n_{i,j} \sim \mathcal{N}\left(0, \sigma^2\right)$. We selected σ^2 to maintain a signal to noise ratio of 10.

Our domain of interest is characterized by high dimensional feature variables and few samples. Hence we set the dimensions as $N = 50$, $D = 200$ and $K = 10$. we performed experiments in the low rank regime (rank $= 2$) and the full rank regime (rank $= 10$). Experiments were performed using training, validation and test sets with the same number of samples. All experiments were repeated 10 times. The validation set was used for parameter selection. The regularization parameter for all the models except for PLS were selected from the set $\{10^{-3}, 10^{-2}, \ldots 10^{3}\}$. PLS is not regularized but requires selection of the number of factors. These were chosen from the set $\{2, 4, \ldots 10\}$.

The regression accuracy was measured using the coefficient of determination on the test set. The R^2 metric given by $1 - \sum(\hat{y} - y)^2 / \sum(y - \mu)^2$ where y is the target response with sample mean μ and $\hat{y}$ is the predicted response. R^2 measures the gain in predictive accuracy compared to a mean model and has a maximum value of 1. The structure recovery was measured using the area under the *roc* curve (AUC) (Cortes & Mohri, 2004) using the structure of the true precision matrix as the binary target, and the values in


(a) Ground truth


(b) Glasso


(c) MVG

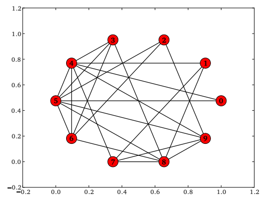

(d) MVG_{rank}

Figure 2: Ground Truth and Recovered Precision Structure with Rank 2 Simulated Data.

Table 1: Average (std.) Accuracy (R^2) and Structure Recovery (AUC) for Rank 2 Simulated Data.

MODEL	R^2	AUC
GLasso	–	0.610 (0.071)
Ridge	0.219 (0.029)	–
PLS	0.214 (0.033)	–
Nuc. Norm	0.220 (0.035)	–
MVG*	0.215 (0.033)	–
$\text{MVG}_{\text{corr.}}$*	0.271 (0.038)	–
MVG_{rank}*	0.296 (0.038)	–
MVG	0.220 (0.035)	0.646 (0.048)
$\text{MVG}_{\text{corr.}}$	0.221 (0.035)	0.648 (0.069)
MVG_{rank}	0.299 (0.038)	0.665 (0.064)

Table 2: Average (std.) Accuracy (R^2) and Structure Recovery (AUC) for Rank 10 Simulated Data.

MODEL	R^2	AUC
GLasso	–	0.708 (0.071)
Ridge	0.180 (0.036)	–
PLS	0.169 (0.037)	–
Nuc. Norm	0.168 (0.037)	–
MVG*	0.179 (0.042)	–
$\text{MVG}_{\text{corr.}}$*	0.246 (0.035)	–
MVG_{rank}*	0.246 (0.035)	–
MVG	0.172 (0.037)	0.702 (0.068)
$\text{MVG}_{\text{corr.}}$	0.172 (0.038)	0.721 (0.071)
MVG_{rank}	0.245 (0.033)	0.700 (0.052)

the recovered precision matrix as scores. AUC measures the quality of the ranking recovered by by the estimated precision matrix. We also present inference only results with the proposed models using the known precision matrix. The results from the simulated data experiments are shown in Table 1 and Table 2. In both tables, we note that (*) represents inference only results using the true precision matrix.

Regression: We found that that accounting for the prior correlation structure had a significant effect on the quality of the recovered regression. Hence, although the Nuc. Norm model performed better than Ridge, models that combined regression with structure recovery outperformed models using regression only. The corrected $\text{MVG}_{\text{corr.}}$ outperformed the MVG in regression suggesting the importance of capturing the posterior covariance for regression and parameter estimation performance. We note that even when the rank is full, the underlying weight matrix is given by the product of the factors is not Gaussian distributed. This may account for the observation that the Gaussian based models perform worse for the full rank data. Another reason may be the significant increase in the effective dimensionality of the weight matrix parameter to be estimated using the same amount of data. MVG_{rank} is able to compensate for this mismatch.

Structure recovery: Overall, all models improved accuracy of recovery for the precision structure as the rank was increased. At the low rank, the MVG_{rank} model was the most effective for structure recovery, but the $\text{MVG}_{\text{corr.}}$ model was the most effective at high rank. We also found that correcting the inference procedure improved the structure recovery performance by comparing $\text{MVG}_{\text{corr.}}$ to MVG . We counted the number of times each edge was selected over the random repetitions. We present the recovered graphs for rank 2 simulated data showing links selected in at least 70% of the repetitions with weight greater than 10^{-6} in Fig. 2. MVG selects many more edges than the $\text{MVG}_{\text{corr.}}$ method in this experiment.

5.2 FUNCTIONAL NEUROIMAGING DATA

Functional magnetic resonance imaging (fMRI) is an important tool for non-invasive study of brain activity. Most fMRI studies involve measurements of blood oxygenation (which are sensitive to the amount of local neuronal activity) while the participant is presented with a stimulus or cognitive task. Neuroimaging signals are then analyzed to identify which brain regions exhibit a systematic response to the stimulation, and thus to infer the functional properties of those brain regions. Functional neuroimaging datasets typically consist of a relatively small number of correlated high dimensional brain images. Hence, capturing the inherent structural properties of the imaging data is critical for robust inference.

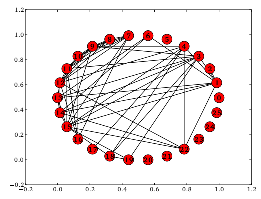
We completed experiments using brain image data from an extended set of the openfMRI database[2]. The data was preprocessed using a general linear model with FMRIB Software Library (FSL) to compute contrast images for each subject resulting in $N = 479$ contrast images for $K = 26$ contrasts. The target contrasts were encoded into a response matrix using the 1-of-k representation, where $y_{n,k} = 1$ if image n corresponds to task k and is zero otherwise. after masking, we are left with $D = 174264$ dimensions. Each dimension in the brain image corresponds to a spatial location in the brain. We used the normalized Laplacian of the 3-dimensional spatial graph of the brain image voxels to define the row precision matrix. This corresponds to the observation that nearby voxels tend to have similar functional activation. Our approach is motivated by the observation that functional neuroimages are highly correlated for different tasks (Poldrack, 2011). We seek to extract this correlation structure as encoded in the prior precision matrix. In addition, the high dimensionality and the similarity between different tasks suggests that the optimal weight matrix may be of low rank.

We divided the training data into five sets using a stratified cross validation to ensure that each training set contains a similar relative number of images corresponding to each task. In addition to the proposed models, we present experimental results using the support vector machine classifier (SVM) using implementations from the *scikit-learn* python package (Pedregosa et al., 2011). We also note that the ridge regression is exactly equivalent to the least square support vector machine (LS-SVM) (Ye & Xiong, 2007) with a linear kernel. For all models (except for PLS), we selected the regularization parameter from the set $\{10^{-3}, 10^{-2}, \dots 10^{3}\}$. The number of factors in PLS was selected from the set $\{2, 4, 6 \dots 26\}$. The results are provided in Table 3.

Table 3: Average (std.) Classification Accuracy for fMRI Data

MODEL	**ACCURACY**
SVM	0.463 (0.052)
PLS	0.422 (0.030)
LS-SVM	0.472 (0.040)
Nuc. Norm	0.234 (0.022)
MVG	0.463 (0.052)
MVG$_{\text{corr.}}$	0.476 (0.050)
MVG$_{\text{rank}}$	**0.512 (0.034)**

[2]`https://openfmri.org/`, extended data provided courtesy of openfMRI.


(a) MVG

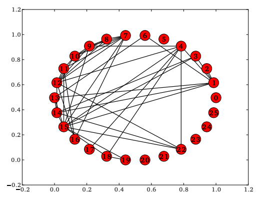

(b) MVG$_{\text{corr.}}$

Figure 3: Recovered Precision Structure for fMRI Data

We note the difficulty of this classification task due to the large number of classes and the high dimensionality of the features. Fig. 3 compares the most significant edges recovered by the precision matrices of MVG and MVG$_{\text{corr.}}$ methods. The figure shows edges with absolute value of the weight greater than the 90^{th} percentile. We found that MVG selected more edges than the MVG$_{\text{corr.}}$ method. We are in the process of collaborating with domain experts for further analysis of the task similarities encoded by the the task precision matrices. These results will be included in an extended version of the paper.

6 CONCLUSION

We proposed a novel primal approach for Bayesian inference subject to possibly non-convex constraints. We applied the proposed inference approach to rank constrained multitask learning. Our approach was motivated by an application to reverse inference for high dimensional functional neuroimaging data. We developed an algorithm for constrained inference that accounts for the latent structure of the predictive weight matrix and constrained parameter estimation to learn the sparse conditional independence structure between the tasks as encoded by the prior precision matrices. We presented experimental performance results compared to strong baseline models on simulated data and real functional neuroimaging data.

We are interested in extending the proposed approach to constrained inference for nonparametric Bayesian models. In particular, we are interested in rank constrained models for the matrix-variate Gaussian process applied to matrix completion. We are also interested in further theoretical development to understand the trade-offs of constrained Bayesian inference compared to other approaches.

Acknowledgments

Authors acknowledge support from NSF grant IIS 1016614. We thank Russell A. Poldrack for providing the extended openfMRI dataset and for help with preprocessing. We also thank Sreangsu Acharyya and Suriya Gunasekar for helpful suggestions on early drafts of this manuscript.

References

Abdi, Herv. Partial least squares regression and projection on latent structure regression. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1), 2010.

Allen, Genevera I. and Tibshirani, Robert. Inference with transposable data: modelling the effects of row and column correlations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2012.

Altun, Yasemin and Smola, Alexander J. Unifying divergence minimization and statistical inference via convex duality. In *COLT*, 2006.

Argyriou, Andreas, Evgeniou, Theodoros, and Pontil, Massimiliano. Multi-task feature learning. In *NIPS*. 2007.

Bishop, Christopher M. Bayesian PCA. In *NIPS*, 1998.

Bishop, C.M. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

Borwein, J.M. and Zhu, Q.J. *Techniques of Variational Analysis*. CMS Books in Mathematics. Springer, 2005.

Brown, L.D. *Fundamentals of Statistical Exponential Families: With Applications in Statistical Decision Theory*. Ims Lecture Notes-Monograph Ser.: Vol.9. Inst of Mathematical Statistic, 1986.

Candés, Emmanuel J. and Recht, Benjamin. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

Cortes, Corinna and Mohri, Mehryar. Auc optimization vs. error rate minimization. In *Advances in Neural Information Processing Systems*. MIT Press, 2004.

Friedman, Jerome, Hastie, Trevor, and Tibshirani, Robert. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Ganchev, Kuzman, Graça, João, Gillenwater, Jennifer, and Taskar, Ben. Posterior regularization for structured latent variable models. *J. Mach. Learn. Res.*, 11:2001–2049, 2010.

Gelfand, Alan E., Smith, Adrian F. M., and Lee, Tai-Ming. Bayesian analysis of constrained parameter and truncated data problems using gibbs sampling. *Journal of the American Statistical Association*, 87(418):pp. 523–532, 1992.

Jaakkola, Tommi, Meila, Marina, and Jebara, Tony. Maximum entropy discrimination. In *NIPS*, 1999.

Kullback, Solomon. *Information Theory and Statistics*. Dover, 1959.

Li, Lu and Toh, Kim-Chuan. An inexact interior point method for l 1-regularized sparse covariance selection. *Mathematical Programming Computation*, 2(3): 291–315, 2010.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.*, 12:2825–2830, 2011.

Pereira, Francisco, Mitchell, Tom, and Botvinick, Matthew. Machine learning classifiers and fMRI: A tutorial overview. *NeuroImage*, 45:S199–S209, 2009. Mathematics in Brain Imaging.

Poldrack, Russell J.A. Inferring mental states from neuroimaging data: From reverse inference to large-scale decoding. *Neuron*, 72(5):692–697, 2011.

Rai, Piyush and Daumé III, Hal. Infinite predictor subspace models for multitask learning. In *AISTATS*, 2010.

Salakhutdinov, Ruslan and Mnih, Andriy. Probabilistic matrix factorization. In *NIPS*, volume 20, 2008.

Smola, A. J. and Kondor, I.R. Kernels and regularization on graphs. In *COLT*, 2003.

Stegle, Oliver, Lippert, Christoph, Mooij, Joris M., Lawrence, Neil D., and Borgwardt, Karsten M. Efficient inference in matrix-variate gaussian models with observation noise. In *NIPS*, 2011.

Williams, Peter M. Bayesian conditionalisation and the principle of minimum information. *The British Journal for the Philosophy of Science*, 31(2):131–144, 1980.

Ye, Jieping and Xiong, Tao. Svm versus least squares svm. In *AISTATS*, 2007.

Yuan, Ming, Ekici, Ali, Lu, Zhaosong, and Monteiro, Renato. Dimension reduction and coefficient estimation in multivariate linear regression. *J. Roy. Stat. Soc. B*, 69(3): 329–346, 2007.

Zellner, Arnold. Optimal information processing and bayes's theorem. *The American Statistician*, 42(4):pp. 278–280, 1988.

Zhang, Yi and Schneider, Jeff G. Learning multiple tasks with a sparse matrix-normal penalty. In *NIPS*, 2010.

Zhu, Jun. Max-margin nonparametric latent feature models for link prediction. In *ICML*, 2012.

Zhu, Jun, Ahmed, Amr, and Xing, Eric P. Medlda: maximum margin supervised topic models for regression and classification. In *ICML*, 2009.

Zhu, Jun, Chen, Ning, and Xing, Eric P. Infinite Latent SVM for Classification and Multi-task Learning. In *NIPS*, 2011.

Zhu, Jun, Chen, Ning, and Xing, Eric P. Bayesian inference with posterior regularization and infinite latent support vector machines. *CoRR*, abs/1210.1766, 2012. Retrieved 02/01/2013.

Collective Diffusion Over Networks: Models and Inference

Akshat Kumar
Analytics and Optimization
IBM Research India
akkumaro@in.ibm.com

Daniel Sheldon
Dept. of Computer Science
University of Massachusetts Amherst
sheldon@cs.umass.edu

Biplav Srivastava
Analytics and Optimization
IBM Research India
sbiplav@in.ibm.com

Abstract

Diffusion processes in networks are increasingly used to model the spread of information and social influence. In several applications in computational sustainability such as the spread of wildlife, infectious diseases and traffic mobility pattern, the observed data often consists of only *aggregate* information. In this work, we present new models that generalize standard diffusion processes to such collective settings. We also present optimization based techniques that can accurately learn the underlying dynamics of the given contagion process, including the hidden network structure, by only observing the time a node becomes active and the associated aggregate information. Empirically, our technique is highly robust and accurately learns network structure with more than 90% recall and precision. Results on real-world flu spread data in the US confirm that our technique can also accurately model infectious disease spread.

1 Introduction

Dynamic phenomena such as the spread of information, ideas, and opinions (Domingos and Richardson, 2001; Kempe *et al.*, 2003; Leskovec *et al.*, 2007) can be described as a *diffusion* process or *cascade* over an underlying network. Similar diffusion processes have also been used for *metapopulation* modeling in the ecology literature to describe how wildlife spreads over a fragmented landscape (Hanski, 1999) and to describe infectious disease propagation among humans (Anderson and May, 2002; Halloran *et al.*, 2010). Such models are crucial for several decision making problems in computational sustainability such as in spatial conservation planning that addresses the question of how to allocate resources to maximize the population spread of an endangered species over a period of time (Sheldon *et al.*, 2010; Ahmadizadeh *et al.*, 2010; Golovin *et al.*, 2011; Kumar *et al.*, 2012).

A fundamental problem in using such diffusion-based models is the estimation of parameters, including the hidden network structure, that govern the contagion process. Recent progress had been made in learning the diffusion parameters of social networks (Myers and Leskovec, 2010; Gomez-Rodriguez *et al.*, 2012; Netrapalli and Sanghavi, 2012; Wang *et al.*, 2012). Myers and Leskovec (2010) formulate the problem of network structure learning as a separable convex program. Gomez-Rodriguez *et al.* (2012) address the problem using submodular optimization. Netrapalli and Sanghavi (2012) address the complementary question of how many observed cascades are necessary to correctly learn the structure of a network. Wang *et al.* (2012) enrich the structure learning problem using additional features from Twitter data.

An implicit assumption commonly made in previous approaches in the social network setting is that one can track each individual in the network and exploit this information during inference. However, in several computational sustainability domains such as ecology, social sciences and transportation, data identifying a single individual is rarely available. For example, for population modeling of migratory birds, one may only know the total number of birds present in a geographical area. While modeling the spread of infectious diseases, one may only know the total number of infected individuals in a community. Similarly, traffic data usually takes the form of vehicle counts leaving or entering an intersection. In theory, one can model such aggregate behavior by explicitly reasoning about each individual in the population. However, such a model cannot be scaled to large population sizes.

We therefore present new *collective diffusion* models that can reason with the aggregate data without the need to model individual-level behavior. These models generalize the well known diffusion models such as the independent cascade model (Kempe *et al.*, 2003). We show how such models can be used to analyze the population dynamics of wildlife and spread of contagious diseases. We also present a model that can address collective diffusion in domains such as transportation that do not fall under the independent cascade model. We highlight how this model is closely related to the recently developed class of collective graphical models (CGMs) (Sheldon and Dietterich, 2011). Such connections are attractive as they open the door to the application of efficient inference techniques developed for CGMs to transportation domain.

The primary algorithmic contribution of our work is to develop algorithms for learning in collective diffusion models using observed data about the infection times of nodes and the associated aggregate information. Using scalable techniques based on convex optimization, we show that our approach can accurately learn network structure with more than 90% precision and recall on large synthetic benchmarks even with a limited number of observed cascades. Furthermore, our approach can also learn edge strength parameters accurately, with less than 2% error. Results on real-world flu spread data available from Centers for Disease Control and Prevention (CDC) in the US confirm that our technique can also accurately model infectious disease spread.

2 Diffusion Over Networks

We first introduce the well known *independent cascade* model, also called the *Susceptible-Infected* (SI) model, for diffusion over a network (Kempe *et al.*, 2003) and later present its collective variants. The main steps in this model are the following:

- We start with an initial set S_0 of active nodes called *seeds* in the network. The process then unfolds in discrete time steps.
- When a node v first becomes active, it is given a single chance to activate each of its currently inactive neighbors w. It succeeds with probability p_{vw} independently of the history of activations so far. Whether the node v succeeds or fails in activating the node w, it cannot make further attempts to activate w in the future.
- If a currently inactive node w has multiple newly active neighbors, their activation attempts of w are sequenced arbitrarily.
- A node w, once active, remains active for the entire diffusion process.

There are several extensions of the basic diffusion model above such as those allowing the nodes to re-

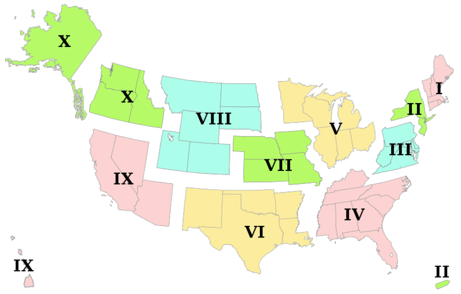


Figure 1: The US map showing 10 federal regions

cover and become infected again. It is easy to fold such extensions into the basic SI model using a time-indexed layered graph (Kempe *et al.*, 2003). A crucial inference problem in such a setting is estimating the edge activation probabilities p_{vw}. The edge activation probabilities also identify the connectivity structure of the network—if $p_{vw} = 0$, then there is no directed edge from node v to w. Next, we describe the observation model commonly used to address this parameter learning problem.

Observation model: A cascade c over such a network starts with a set of initially active nodes at time t=0. As the cascade progresses in discrete time steps, we observe the infection time τ^c of nodes as they subsequently become infected; for nodes u that are never infected, we set the infection time $\tau^c_u = \infty$. Furthermore, for the activated nodes, we do not observe which node activated them. Therefore, the connectivity structure of the network is hidden. There exist a number of techniques that can estimate the parameters p_{vw} for each edge using this observation model (Myers and Leskovec, 2010; Gomez-Rodriguez *et al.*, 2012; Netrapalli and Sanghavi, 2012; Wang *et al.*, 2012).

3 Collective Diffusion—CSI Model

The typical observation model used in the social networking setting assumes that one can track the status of each individual in the network and exploit this information during inference. However, this assumption rarely holds in several computational sustainability domains such as ecology, social sciences and transportation, where data identifying a single individual is not often available. We motivate this observation through the following examples.

In wildlife population modeling, the goal is to describe the occupancy pattern of habitat patches for a certain species in a fragmented landscape over a period of time (Hanski, 1999; Sheldon *et al.*, 2010). Each habitat patch i can be thought of as a node in a geospatial network. A habitat patch i can provide support for at most N_i members of a species. This model works

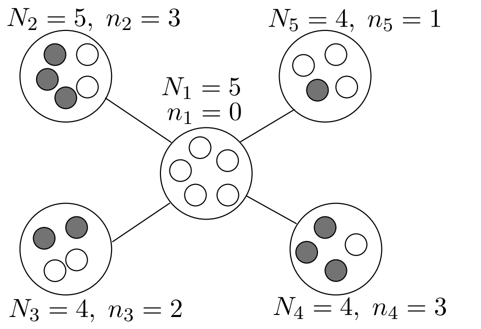


Figure 2: An example of collective diffusion in a 5-node network. Each small circle represents an individual within the larger node. The total population of a node is given as N_i; number of active individuals within a node shown in grey circles is denoted as n_i

well for many animals, but is particularly appropriate for territorial species, in which an individual or a family group defends a distinct territory within the patch for breeding and foraging. A concrete example is a species of cavity-nesting bird such as the Eastern Bluebird, who do not excavate their own cavities, but rely on those made by other species. In this case, the number N_i corresponds to the number of available nest cavities. As it is difficult to track individual birds, the observed data often consists of only the number of species present in a habitat patch i, say n_i. Based on this *aggregate* observed data, we need to infer the *colonization* probability p_{vw} that represents the probability that an individual from patch v will colonize an unoccupied cavity in patch w.

Similar collective diffusion settings arise while modeling the spread of infectious diseases (Abbey, 1952; Halloran *et al.*, 2010). The region under observation is divided into multiple sub-communities. For example, the CDC in the United States reports flu data for 10 federal health regions as shown in Fig. 1. The observed data consists of the total number of infected individuals in a sub-community and the time data is collected (the particular week of the year). Therefore, the need to model and reason with *aggregate* data motivates the development of the following *Collective-Susceptible-Infected* (CSI) model.

3.1 The CSI Model

In the CSI model, we are given a graph $G = (V, E)$. Each node i in the graph represents a sub-community of individuals. A node i can support a maximum of N_i individuals. Each individual can be either *active* or *inactive*. A complete observed cascade c is the collection of nodes and the infection time τ_i and the number of individuals n_i that are activated for each node i. For simplicity, we refer to n_i as node i's *activation level.* We call a node i active if $n_i \geq 1$. That is, it has at least one active individual. A CSI cascade proceeds in a similar manner as described in Sec. 2:

- We start with an initial set S_0 of active nodes called *seeds* in the network. Each active node has an activation level $1 \leq n_i \leq N_i$. The process then unfolds in discrete time steps.
- When a node j first becomes active, it is given a single chance to activate each currently inactive neighbor i. Each *active* individual in node j can activate an *inactive* individual in node i with probability p_{ji}. Whether node j succeeds or fails in activating any individual in node i, it cannot make further attempts to activate i in the future.
- If a currently inactive node i has multiple newly active neighbors, their activation attempts of i are sequenced arbitrarily.
- Node i, once active with activation level n_i, remains active with the same activation level for the entire diffusion process.

As also highlighted in Sec. 2, we can model *non-progressive* cascades in which the activation level of nodes change, such as the changing population of species in a habitat patch with time, by using the above diffusion process in a time-indexed layered graph (Kempe *et al.*, 2003).

A critical issue to address in such a collective diffusion process is to address how many individuals become active in a currently inactive node i. Let us assume that the current time step is t. Consider a single individual i_m within the node i. Let $X(t)$ denote the set of all newly activated nodes at time t: $X(t) = \{i \in V : \tau_i = t\}$. The probability that i_m is not active given the status of its neighbors is given as:

$$P\big(i_m = 0 \text{ at time } t \,|\, X(t-1)\big) = \prod_{j \in X(t-1)} \big(1 - p_{ji}\big)^{n_j}$$

Therefore, the probability that individual i_m is active is given as:

$$P\big(i_m = 1 \text{ at time } t \,|\, X(t-1)\big) = 1 - \prod_{j \in X(t-1)} \big(1 - p_{ji}\big)^{n_j}$$

As the individuals within the node i are identical, one can think of the process of determining the number of active individuals n_i within the node i as sampling from the following Binomial distribution:

$$P\big(n_i = \text{active at time } t \,|\, X(t-1)\big) = \frac{N_i!}{n_i!(N_i - n_i)!} \left(1 - \prod_{j \in X(t-1)} \big(1 - p_{ji}\big)^{n_j}\right)^{n_i} \prod_{j \in X(t-1)} \big(1 - p_{ji}\big)^{n_j(N_i - n_i)} \tag{1}$$

Based on this understanding of the CSI model, we are now ready to describe the maximum likelihood formulation of the parameter learning problem.

3.2 Parameter Learning for CSI Model

Let $\mathcal{C}$ denote the set of all observed cascades. For each cascade $c \in \mathcal{C}$, we only observe which nodes are active at each time step and their activation level. We do not observe how a node got infected, which particular individuals within a node are active/inactive or the underlying connectivity structure of the network. The goal is to learn the parameters p_{ij} for each pair of nodes $i, j \in V$. Our approach is based on maximizing the likelihood of the observed data. Similar maximum likelihood (ML) based approaches have been used in (Myers and Leskovec, 2010; Netrapalli and Sanghavi, 2012). However, previous approaches are not applicable to the collective variant which we address.

Let A denote the matrix of activation probabilities. Let c denote a particular cascade $c \in \mathcal{C}$. Let $\mathcal{I}^c$ denote the set of nodes that become activated at some point in cascade c; $\mathcal{U}^c$ denote the nodes that remain unactivated. The probability of the observed cascades is:

$$P(\mathcal{C}; A) = \prod_{c \in \mathcal{C}} \Bigg[\Bigg(\prod_{i \in \mathcal{I}^c} P\big(n_i^c \text{ active at time } \tau_i^c | X^c(\tau_i^c - 1)\big)$$
$$P\big(\text{Node } i \text{ inactive before time } \tau_i^c | X^c(t < \tau_i^c - 1)\big) \Bigg)$$
$$\Bigg(\prod_{i \in \mathcal{U}^c} P\big(\text{Node } i \text{ always inactive }\big) \Bigg) \Bigg] \quad (2)$$

where $X^c(t < \tau_i - 1)$ denotes the set of all nodes that were activated before time $\tau_i^c - 1$ in a cascade c. The first term in the above expression is given in Eq. (1). We write the expressions for the remaining two terms for a particular cascade c as follows:

$$P\big(\text{Node } i \text{ inactive before time } \tau_i^c | X^c(t < \tau_i^c - 1)\big) = \prod_{j \in V: \tau_j^c < \tau_i^c - 1} (1 - p_{ji})^{n_j^c N_i} \quad (3)$$

The probability that node i never became activated is similarly given as:

$$P\big(\text{Node } i \text{ always inactive}\big) = \prod_{j \in V: \tau_j^c < \infty} (1 - p_{ji})^{n_j^c N_i} \quad (4)$$

The maximum likelihood problem entails finding the activation probability matrix A that maximizes the following:

$$\max_A \log P(\mathcal{C}; A) \quad (5)$$

We can easily see that maximizing the above log-likelihood can be performed independently for each node i in the network, which makes the approach highly scalable. Therefore, the optimization problem for a particular node i is given as:

$$\max_{\{p_{ji}\}} \sum_{c \in \mathcal{C}: \tau_i^c < \infty} n_i^c \log \Bigg(1 - \prod_{j \in X^c(\tau_i^c - 1)} (1 - p_{ji})^{n_j^c} \Bigg) +$$
$$\sum_{c \in \mathcal{C}: \tau_i^c < \infty} \sum_{j \in X^c(\tau_i^c - 1)} n_j^c (N_i - n_i^c) \log \big(1 - p_{ji}\big) +$$
$$\sum_{c \in \mathcal{C}: \tau_i^c < \infty} \sum_{j \in V: \tau_j^c < \tau_i^c - 1} n_j^c N_i \log \big(1 - p_{ji}\big) +$$
$$\sum_{c \in \mathcal{C}: \tau_i^c = \infty} \sum_{j \in V: \tau_j^c < \infty} n_j^c N_i \log \big(1 - p_{ji}\big) \quad (6)$$

The above optimization is not convex and thus, direct optimization may not produce optimal solutions. We next show how to make the above problem convex by using a change of variables similar in spirit to the approach in (Myers and Leskovec, 2010). Let us introduce the following substitutions:

$$q_{ji} = 1 - p_{ji} \quad (7)$$
$$\gamma_i^c = 1 - \prod_{j \in X^c(\tau_i^c - 1)} q_{ji}^{n_j^c} \quad (8)$$
$$\hat{q}_{ji} = \log q_{ji} \quad (9)$$
$$\hat{\gamma}_i^c = \log \gamma_i^c \quad (10)$$

The new equivalent optimization problem is given as:

$$\max_{\{\hat{q}_{ji}, \hat{\gamma}_i^c\}} \sum_{c \in \mathcal{C}: \tau_i^c < \infty} \Bigg\{ n_i^c \hat{\gamma}_i^c + \sum_{j \in X^c(\tau_i^c - 1)} n_j^c (N_i - n_i^c) \hat{q}_{ji}$$
$$+ \sum_{j \in V: \tau_j^c < \tau_i^c - 1} n_j^c N_i \hat{q}_{ji} \Bigg\} + \sum_{c \in \mathcal{C}: \tau_i^c = \infty} \sum_{j \in V: \tau_j^c < \infty} n_j^c N_i \hat{q}_{ji} \quad (11)$$

$$\text{s.t. } \exp\big(\hat{\gamma}_i^c\big) + \exp\Big(\sum_{j \in X^c(\tau_i^c - 1)} n_j^c \hat{q}_{ji}^c \Big) \le 1 \; \forall c : \tau_i^c < \infty \quad (12)$$

$$\hat{q}_{ji} \le 0 \; \forall j \in V \quad (13)$$
$$\hat{\gamma}_i^c \le 0 \; \forall c : \tau_i^c < \infty \quad (14)$$

In the above problem, the objective function is linear in all the variables. We have represented the equality constraint in Eq. (8) using the inequality constraint (12). This is justified as the objective function is an increasing function of both $\hat{\gamma}_i^c$ and $\hat{q}_{ji}$. Therefore, at the optimal solution, there will be no slack for this constraint and Eq. (8) will hold. To make constraint (12) convex, we take log of both the sides and get an equivalent convex constraint as:

$$\log \Bigg(\exp\big(\hat{\gamma}_i^c\big) + \exp\Big(\sum_{j \in X^c(\tau_i^c - 1)} n_j^c \hat{q}_{ji}^c \Big) \Bigg) \le 0 \quad (15)$$

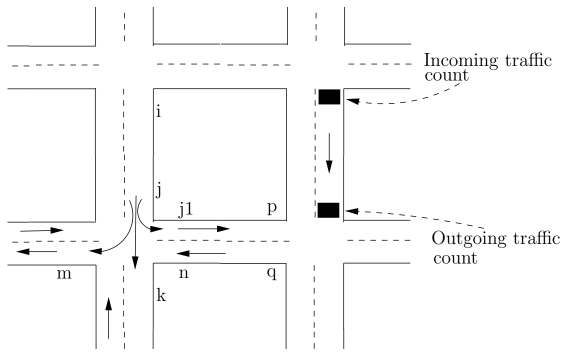


Figure 3: A road network showing data collection methodology using loop detectors that can count incoming and outgoing vehicles for a road segment

Now the optimization problem (11) is a convex problem subject to constraints (13,14,15) and therefore, can be solved optimally using off-the-shelf nonlinear solvers such as SNOPT.

As also noted in previous work (Myers and Leskovec, 2010), networks are generally sparse. To encourage sparsity, we add the following sparsity inducing penalty term to the objective function for a node i, with parameter $\rho > 0$:

$$-\rho \sum_{j \in V} e^{-\hat{q}_{ji}} \tag{16}$$

The above penalty term accurately predicts edges which are not part of the underlying network. That is, $p_{ji}=0$ for such edges. However, a side effect of the penalty term is that it skews the true parameters p_{ji} for other edges. Therefore, once the underlying structure of the network is discovered using this penalty term, we solve the optimization problem again to accurately recover the true parameters p_{ji}.

4 Collective Flow Diffusion Model

In the collective diffusion model of the previous section, the total number of activated individuals and nodes increases as the underlying contagion spreads through the network. In applications such as traffic flow modeling, given certain input traffic through network entry points, one is interested in modeling how the traffic flow diffuses through the road network. Thus, *flow conservation* is observed for each node of the network. To address such scenarios, we present the *collective flow diffusions* (CFD) model. The CFD model can also be interpreted as a Markov chain, and as highlighted in later sections, is a special case of collective graphical models (CGMs) (Sheldon and Dietterich, 2011). This connection allows for adapting inference strategies for CGMs to the CFD model.

$$\begin{tikzcd}
 & i \arrow[d] & & \\
 & j \arrow[r] \arrow[dl] \arrow[d] & j1 \arrow[r] & p \\
m & k & n \arrow[l] \arrow[ll] & q \arrow[l]
\end{tikzcd}$$

Figure 4: Equivalent network representation for the road network shown in Figure 3, includes only marked locations.

Consider a road network as shown in Fig. 3. A key learning problem in such traffic networks is estimating the *turn probabilities* for each road segment of this network. That is, given a road segment (i, j) as shown in Fig. 3, we want to estimate what fraction of outgoing traffic from location j goes straight, turns right and turns left over a period of time T. Turn probabilities are essential to model the traffic flow in several traffic simulators (Nguyen *et al.*, 1997; Thiebaux *et al.*, 1999) and are a crucial measure that determine the macroscopic properties of the traffic flow such as the congestion level, origin-destination matrix, among others. Several popular analytical models of traffic flow such as the cell transmission model (Daganzo, 1994) are based on the assumption that turn probabilities are known a priori for each location.

In several urban traffic networks, aggregate data in the form of vehicle count is already collected for each road segment using inductive-loop traffic detectors. The main data requirement in our work is the availability of aggregate *incoming* and *outgoing* traffic for a road segment for the total time duration T. For example, black rectangles in Figure 3 show the places where we require aggregate vehicle count. This assumption can be relaxed in principle by treating unavailable vehicle counts as missing data. For now, we handle the simpler case where such traffic counts are available for each road segment. Note that determining turn probabilities from this data is not trivial as we do not observe how much traffic is forwarded to each adjacent link.

4.1 Network and Data Representation

We present the CFD model in the context of traffic networks, but this model applies to any setting where flow is conserved. Each location in the road network is a node in our graph representation. For example, we create one node for each location j, k and m, among others, for the road network in Figure 3. *Directed edges* model road links along with traffic direction. For example, there are directed edges $(j, j1)$, (j, k) and (j, m). The node $j1$ has a single outgoing link to node p for the example in Figure 3. We call location nodes which receive incoming traffic, such as nodes i, $j1$ and m as *inflow* nodes. The nodes where outgoing traffic count is recorded, such as node j and p, are called *outflow* nodes. Figure 4 shows a part of the network representation for the road network in Figure 3.

Observed Data: We observe, for each *inflow* node i in the network, the total incoming traffic count $n_T(i)$ after T time steps. For each *outflow* node o, we observe the total outgoing traffic count $n_{T-1}(o)$ after $T-1$ time steps.

4.2 Complete Data Likelihood

We have the following flow conservation relations for different nodes in the network:

$$n_T(i) = \sum_{o \in \mathrm{Nb}(i)} n_{T-1}(o,i) \tag{17}$$

where $\mathrm{Nb}(i)$ denotes adjacent *outflow neighbors* o of the inflow node i that send a total of $n_{T-1}(o,i)$ vehicles to node i after $T-1$ time steps. In this section, we are assuming that $n_{T-1}(o,i)$ is also observed; the results for this simpler case will pave the way an outline of the Expectation-Maximization algorithm in the next section for the case when only total incoming and outgoing counts are observed. Let $\mathcal{O}$ denote the set of all outflow nodes in the network and $\mathcal{I}$ denote the set of inflow nodes. For each outflow node o, the flow conservation is given as:

$$n_{T-1}(o) = \sum_{i \in \mathrm{Nb}'(o)} n_{T-1}(o,i) \tag{18}$$

where $\mathrm{Nb}'(o)$ denotes adjacent *inflow neighbors* i of the outflow node o that can receive traffic from o. The turn probabilities p_{oi} are defined for each pair (o,i) of adjacent outflow and inflow nodes. They intuitively represent the probability that a vehicle at the node o will turn to node i. The complete data joint probability is given as:

$$P(\boldsymbol{n}; \{p_{oi}\}) = \prod_{o \in \mathcal{O}} \left[\frac{n_{T-1}(o)!}{\prod_{i \in \mathrm{Nb}'(o)} n_{T-1}(o,i)!} \prod_{i \in \mathrm{Nb}'(o)} p_{oi}^{n_{T-1}(o,i)} \right] \tag{19}$$

Subject to the following constraints:

$$\sum_{i \in \mathrm{Nb}'(o)} p_{oi} = 1 \qquad \forall o \in \mathcal{O} \tag{20}$$

$$\sum_{i \in \mathrm{Nb}'(o)} n_{T-1}(o,i) = n_{T-1}(o) \qquad \forall o \in \mathcal{O} \tag{21}$$

$$\sum_{o \in \mathrm{Nb}(i)} n_{T-1}(o,i) = n_T(i) \qquad \forall i \in \mathcal{I} \tag{22}$$

The meaning of the constraints is that the probability is zero when the observed data do not satisfy flow conservation. Intuitively, the expression in Eq. (19) is a product of multinomial distributions, one for each outflow node o, where one can imagine performing $n_{T-1}(o)$ trials which lead to success in exactly one of the categories in the set $\mathrm{Nb}'(o)$. This joint-probability describes a single cascade of flow diffusion and can be easily generalized to multiple independent cascades by using the i.i.d. assumption. Given the complete observed data $\boldsymbol{n}$, estimating the turn probabilities involves solving the following optimization problem subject to constraint (20), which is equivalent to estimating the parameters of a multinomial distribution.

$$\max_{\{p_{oi}\}} \sum_c \log P(\boldsymbol{n}; \{p_{oi}\}) \tag{23}$$

where c denotes a single complete cascade.

4.3 Inference With Missing Data

According to the observation model in Section 4.1, the variables $n_{T-1}(o,i)$ are not observed for any location pairs. Therefore, the approach of Section 4.2 cannot be applied directly. However, recently inference approaches to address missing data in collective graphical model settings have been proposed (Sheldon and Dietterich, 2011). Sheldon and Dietterich (2011) address the problem of generating samples from the posterior distribution of a collective graphical model by deriving an efficient Gibbs sampling algorithms that can work with hard constraints as in (21) and (22). Therefore, such a sampling strategy can be used in conjunction with the EM algorithm (Dempster *et al.*, 1977) to generate complete data samples conditioned on aggregate data for the traffic network. The M-step of the EM algorithm involves a similar optimization as in Eq. (23).

5 Experiments

In this section, we present the results of our inference approach for a number of synthetic and real-world data sets. We focus on the collective diffusion model of Section 3. Our diffusion simulator was implemented in JAVA and used the nonlinear programming solver SNOPT (Gill *et al.*, 2002) with AMPL interface (Fourer *et al.*, 2002) to solve the optimization problem (11). The experiments were run on a Mac Pro with a single 2.4GHz processor and 4GB RAM allocated to the solver.

For synthetic benchmarks, we generated 100, 250 and 500 node scale-free networks with the preferential attachment model similar to (Myers and Leskovec, 2010). The largest 500 node network had about 900

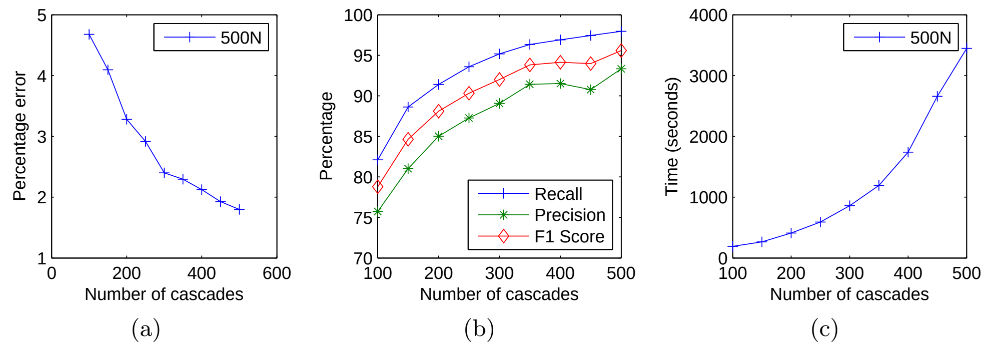


Figure 5: Error, recall, precision and timing results for the 500 node network

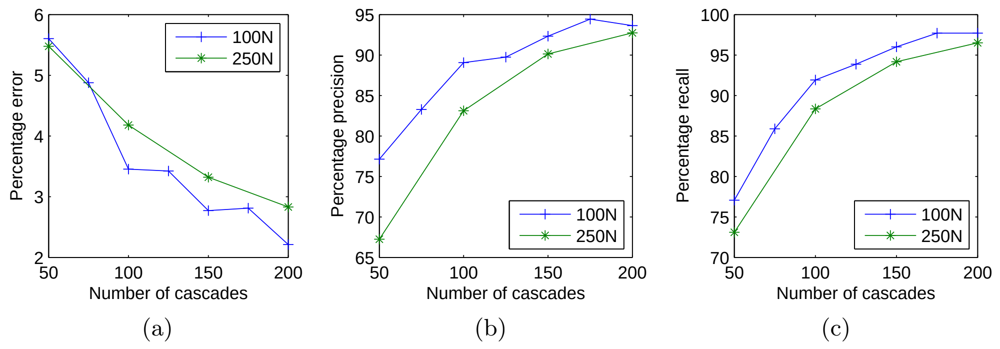


Figure 6: Error, precision and recall results for 100 and 250 node networks

edges. The edges were considered bi-directed, implying 1800 edges for the 500 node network. The edge log-activation probabilities, $\ln p_{ij}$, were sampled uniformly randomly in the range $[-8, -4.6]$ for each run. A main objective of the experiments is to test the efficacy of the optimization approach of Section 3.2 w.r.t. a varying number of cascades. Ideally, one would like to learn the network structure and the edge parameters accurately with as few cascades as possible. Encouragingly, our approach is quite successful in achieving this objective as highlighted next.

Figure 5 shows the results for the largest 500 node network. Each point in the plots is the average of 5 runs. We fixed the maximum population N_i of each node to 1000. To make the inference challenging, 5% of total nodes were initialized as seeds, resulting in 25 seeds for the 500 node network. The activation level of seeds was sampled uniformly from the range $[5, 25]$. The x-axis of each plot represents the number of cascades and the y-axis shows the measured property.

Figure 5(a) shows the percentage error in estimating the edge activation parameter for the set of correctly predicted edges, say S, calculated as:

$$error = 100 * \frac{\sum_{ij \in S} |p_{ij}^{estimate} - p_{ij}^{true}|}{\sum_{ij \in S} p_{ij}^{true}} \tag{24}$$

In this 500 node network, even with 100 cascades, the error is quite small, around 5%. We contrast this with the setting in (Myers and Leskovec, 2010), where roughly the same number of cascades were generated as the number of nodes. Our results show that under the collective diffusion model, one can obtain good results with significantly fewer cascades. As expected, the error decreases as the number of observed cascades increases. For 500 cascades, it is around 2%, resulting in very high accuracy.

Figure 5(b) shows the precision, recall and the F_1 score with varying number of cascades. For 100 cascades, the precision and recall are 75% and 82% respectively. It is encouraging that we can get reasonable results even with very few cascades. Furthermore, both the precision and recall increase sharply w.r.t. the number of cascades. The F_1 score is around 90% for 250 cascades. This shows that our approach is particularly effective in utilizing additional data. For 500 cascades, the F_1 score is already 95% resulting in very high accuracy and precision in estimating the original network.

Figure 5(c) shows the total runtime of our approach which includes the time to generate cascades and solve the optimization problem. The runtime as expected increases w.r.t. the number of cascades. It takes about

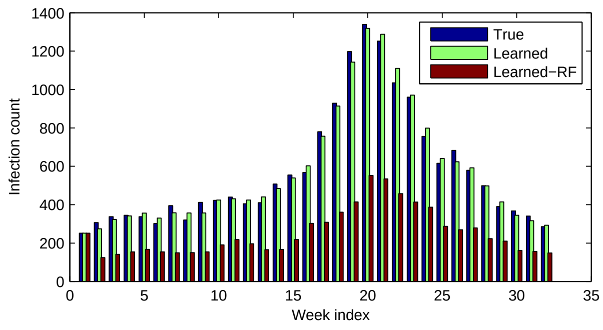


Figure 7: Comparison of true infection count, predicted counts using our approach ('Learned') and predicted counts using the Reed-Frost model ('Learned–RF') for the US Federal region 1

an hour to solve the largest 500 cascade instance. A main contributing factor to the runtime is the sharp increase in the size of the convex program for each node with the number of cascades. The SNOPT solver takes about 6 minutes to solve the optimization problem for each node, the size of which is about 3MB in the AMPL format. However, the good news is that once the cascades are generated, one can drastically reduce the runtime by solving the optimization problem for each node independently. Therefore, there is a dramatic potential for speedup by using cloud computing or multicore machines.

Figure 6 shows the error, precision and recall results for smaller 100 and 250 node networks. The results are similar to those obtained on the 500-node networks. Our approach is able to achieve more than 90% recall and precision for both these cases, further providing a proof its efficacy.

CDC Data: We also tested the CSI model of Section 3 to model the spread of flu in the US for the season 2010-11. The data is made publicly available by the Centers for Disease Control and Prevention (CDC, `http://www.cdc.gov`). The data is available for each of the 10 Federal regions of the US as shown in Figure 1. For each region, the relevant data consists of tuples $\langle$Week number, # of flu patients$\rangle$. We consider the peak of the flu season from week 40 (October) till week 20 (May) of the next year. In this setting, we consider the graph to be fully connected with each of the 10 regions potentially able to influence every other region. As the number of flu patients varies with time, we model it using non-progressive cascades using a time indexed graph, with each layer corresponding to the particular week number. The parameter N_i is the total population of all the states in region i.

The strength of influence of a region i on region j is denoted as p_{ij}. Intuitively, it denotes how region i's flu population influences the flu spread in region j. As no true model describing flu spread is available, we make certain assumptions that attempt to avoid overfitting of the data and represent some intuitions about the disease spread. They are as follows. First, all the variables p_{ij} are independent of the particular time of the year. That is, *inter-region* spread has same parameters for every week. This represents the intuition that travel trends that affect the inter-region spread roughly remain the same throughout the year.

Second, the flu spread probability within a region i (or the *intra-region* spread) for a particular week t is modeled as p^t_{ii} to describe how the flu spread strength varies with the time of the year. This is justified as the flu spread depends on the intensity of the cold weather, which varies with time. Finally, instead of the intra-region spread being independent for each Federal region, we constrain them to be within a certain percentage of a base probability, that itself is an optimization variable. That is:

$$0.8 \leq \frac{p^t_{ii}}{p^t_{\text{ref}}} \leq 1.2\ \forall i,\ \forall t \qquad (25)$$

where p^t_{ref} is the base flu spread probability for a particular time t and is itself an optimization variable. The main effect of this constraint is that it couples the intra-region spread probability of all the regions. This constraint further attempts to avoid the overfitting of data. Finally, the main variables to estimate are the *inter-region* spread probabilities p_{ij} for each pair of 10 regions, the *intra-region* spread probabilities p^t_{ii} for each week t and region i, and the base spread probability p^t_{ref} for each week t.

We also note that a similar model to ours is used to model disease spread (Halloran *et al.*, 2010; Abbey, 1952). In particular, the Reed-Frost (RF) model (Halloran *et al.*, 2010; Abbey, 1952) is very similar to the CSI model. The key advantage of the CSI model is that it can model and learn the influence of nodes on each other. The RF model does not allow such inter-region effects and thus, its parameters are much simpler to estimate.

We first compare the accuracy of our model and the RF model. For the RF model, we include the constraint (25), otherwise it is trivial to fit the data with almost 100% accuracy by adjusting the intra-region to track observed flu intensity, which represents overfitting of data. The results for our model are quite encouraging. The average error in predictions using our model is only 3.8% for all the regions and weeks. The minimum error is 1.15% for region 5. The maximum error is 10% for region 7. For the RF model, the average error is 31%. This confirms our modeling assumption that inter-region spread is crucial to take

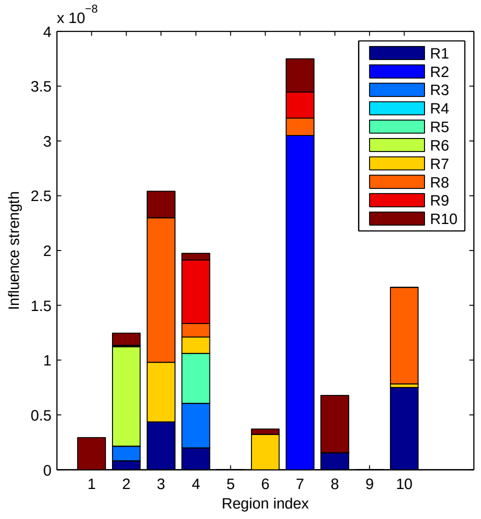


Figure 8: Inter-region influence visualization (best viewed in color). The x-axis denotes a particular region, y-axis denotes the strength of its influence on other regions using the color coding scheme on the right

into account. The accuracy of the RF model, which does not model the inter-region spread, suffers significantly. Figure 7 shows the weekly predictions for our model, the RF model and the true observed data for region 1. We can easily see the difference in the accuracy of our model and the RF model.

Figure 8 shows a visualization of the inter-region influence p_{ij} for all the 10 regions. We note that as the true model for flu spread is not known, we must be careful in interpreting this result. Regardless, this plot depicts several trends that are intuitively correct and provide a justification to some of our modeling assumptions. Regions 2, 3 and 4 are the ones most responsible for spreading the flu to other regions, indicated by the number of stacks for each of them. This is intuitively justified as these regions represent the North-East and the South-East of the US (see Figure 1) and are known to have strong flu season due to intense cold weather. Region 9, which includes California, is generally known to have a weak flu season and this is reflected in Figure 8, where region 9 has zero influence on other regions.

We also note that our post-hoc analysis can be useful for health providers to better prepare for the flu season next year. The model we presented can precisely estimate the strength of flu spread for different regions and at different time of the year. This knowledge can help health care providers to prepare for contingencies for the future flu season. Currently, we only modeled the observed data for a single flu season. Predicting future flu seasons on a weekly or bi-weekly basis based on past season's data remains an important future work and will require additional analysis and inputs.

6 Conclusion

In several computational sustainability applications including the spread of wildlife, infectious diseases and traffic flow, the observed data often consists of only collective information, without any identifiable details about individuals in the population. In our work, we presented models that generalized the standard diffusion models such as the independent cascade model to collective settings. We motivated such collective models based on applications in ecology, infectious disease spread and transportation. We also developed scalable convex optimization based techniques that can accurately learn the parameters, including the hidden structure of the underlying network, by observing only timestamped aggregate data. Experiments on a number of synthetic and real-world benchmarks show that our approach is highly accurate and can recover the hidden structure for large networks with high precision and recall even with limited observed data.

Our future work includes further exploration of inference based techniques for modeling the traffic flow and disease spread. Addressing both these applications can create a significant practical impact. Further investigation of the connections we established between these domains and graphical models and optimization would certainty lead to deeper insights in modeling and decision making for these domains.

Acknowledgements

Daniel Sheldon is supported by the National Science Foundation under Grant No. IIS-1117954.

References

H. Abbey. An examination of the Reed-Frost theory of epidemics. *Human Biology*, 24(3):201–233, 1952.

Kiyan Ahmadizadeh, Bistra Dilkina, Carla P. Gomes, and Ashish Sabharwal. An empirical study of optimization for maximizing diffusion in networks. In *Proceedings of the 16th International Conference on Principles and Practice of Constraint Programming*, pages 514–521, 2010.

Roy M. Anderson and Robert M. May, editors. *Infectious diseases of humans: dynamics and control.* Oxford press, 2002.

C.F. Daganzo. The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269–287, 1994.

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*, 39(1):1–38, 1977.

Pedro Domingos and Matt Richardson. Mining the network value of customers. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 57–66, 2001.

Robert Fourer, David M. Gay, and Brian W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press, November 2002.

Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.

Daniel Golovin, Andreas Krause, Beth Gardner, Sarah J. Converse, and Steve Morey. Dynamic resource allocation in conservation planning. In *Proceedings of the 25th Conference on Artificial Intelligence*, pages 1331–1336, 2011.

Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. *ACM Transactions on Knowledge Discovery From Data*, 5(4):21, 2012.

M. Elizabeth Halloran, Ira Longini, and Claudio Struchiner. Binomial and stochastic transmission models. In *Design and Analysis of Vaccine Studies*, Statistics for Biology and Health, pages 63–84. Springer New York, 2010.

I. Hanski, editor. *Metapopulation ecology*. Oxford University Press, 1999.

David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146, 2003.

Akshat Kumar, Xiaojian Wu, and Shlomo Zilberstein. Lagrangian relaxation techniques for scalable spatial conservation planning. In *AAAI Conference on Artificial Intelligence*, pages 309–315, 2012.

Jure Leskovec, Lada A. Adamic, and Bernardo A. Huberman. The dynamics of viral marketing. *ACM Trans. Web*, 1(1), May 2007.

Seth A. Myers and Jure Leskovec. On the convexity of latent social network inference. In *Advances in Neural Information Processing Systems*, pages 1741–1749, 2010.

Praneeth Netrapalli and Sujay Sanghavi. Learning the graph of epidemic cascades. In *ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '12, pages 211–222, 2012.

V. N. Nguyen, P. Kilby, and P. Lamb. Validation of the TRITRAM traffic network simulation model. In *International Conference of ITS Australia*, 1997.

Daniel R. Sheldon and Thomas G. Dietterich. Collective graphical models. In *Advances in Neural Information Processing Systems*, pages 1161–1169, 2011.

Daniel Sheldon, Bistra Dilkina, Adam Elmachtoub, Ryan Finseth, Ashish Sabharwal, Jon Conrad, Carla Gomes, David Shmoys, William Allen, Ole Amundsen, and William Vaughan. Maximizing the spread of cascades using network design. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, pages 517–526, 2010.

Sylvie Thiebaux, Peter Lamb, and Bella Robinson. Updating turn probabilities at intersections in response to traffic incidents. In *International Conference of ITS Australia*, 1999.

Liaoruo Wang, Stefano Ermon, and John E. Hopcroft. Feature-enhanced probabilistic models for diffusion network inference. In *European conference on Machine Learning and Knowledge Discovery in Databases*, ECML PKDD'12, pages 499–514, 2012.

Causal Transportability of Experiments on Controllable Subsets of Variables: z-Transportability

Sanghack Lee and Vasant Honavar
Artificial Intelligence Research Laboratory
Department of Computer Science
Iowa State University
Ames, IA 50011

Abstract

We introduce *z-transportability*, the problem of estimating the causal effect of a set of variables **X** on another set of variables **Y** in a *target domain* from experiments on any subset of controllable variables **Z** where **Z** is an *arbitrary* subset of observable variables **V** in a *source domain.* z-Transportability generalizes z-identifiability, the problem of estimating in a given domain the causal effect of **X** on **Y** from surrogate experiments on a set of variables **Z** such that **Z** is disjoint from **X**. z-Transportability also generalizes transportability which requires that the causal effect of **X** on **Y** in the target domain be estimable from experiments on any subset of *all* observable variables in the source domain. We first generalize z-identifiability to allow cases where **Z** is not necessarily disjoint from **X**. Then, we establish a necessary and sufficient condition for z-transportability in terms of generalized z-identifiability and transportability. We provide a sound and complete algorithm that determines whether a causal effect is *z-transportable*; and if it is, produces a transport formula, that is, a recipe for estimating the causal effect of **X** on **Y** in the target domain using information elicited from the results of experimental manipulations of **Z** in the source domain and observational data from the target domain. Our results also show that *do*-calculus is complete for z-transportability.

1 INTRODUCTION

Elicitation of a causal effect from observations and experiments is central to scientific discovery, or more generally, rational approaches to understanding and interacting with the world around us. *Causal diagrams* (Pearl, 1995, 2000) provide a formal representation for combining data with causal information. *Do*-calculus (Pearl, 1995, 2000, 2012) provides a sound (Pearl, 1995) and *complete* (Shpitser and Pearl, 2006b; Huang and Valtorta, 2006) inferential machinery for causal inference. The resulting framework has been used to estimate *causal effects* of a set of variables **X** on another set of variables **Y** from *observations* and *interventions* (Pearl, 2000; Tian and Pearl, 2002; Tian, 2004; Shpitser and Pearl, 2006a).

In real world scenarios in which the treatment variables **X** may not be amenable to interventions due to technical or ethical considerations, it is interesting to consider experiments on a possibly different set of variables **Z** that are more amenable to manipulate than the treatment variables **X**. Bareinboim and Pearl (2012a) introduced *z-identifiability,* the problem of estimating in a given domain (setting, environment, population) the causal effect of **X** on **Y** from surrogate experiments on **Z**. In scenarios in which causal information acquired from one domain might be useful another different but related domain. Pearl and Bareinboim (2011) introduced *selection diagrams* for expressing knowledge about differences and commonalities between a source and a target domain. They used the selection diagrams to provide a formal definition of *transportability*, a license to transport causal information elicited from *experimental* studies in a *source* to a *target* domain in which only an *observational* study is possible. They also provided an algorithm for determining whether a causal effect is transportable given a selection diagram that represents a set of assumptions about the differences between the source and the target domains; and if so, computing a *transport formula,* which provides a recipe for estimating a causal effect in the target domain. In transporting the causal effect of a set of variables **X** on another set of variables **Y** from a source to a target domain we are free to use information acquired from *all possible* experiments on any subset of **V** in the source domain, available knowledge about differences and commonalities between the source and target domains (encoded by the selection diagram), and observations in both domains. However, many scenarios of practical interest present the problem of estimating in a target domain the causal effect of a set of variables $\mathbf{X}$ on another set of variables $\mathbf{Y}$ using experiments in the source domain on a subset $\mathbf{Z}$ of $\mathbf{V}$.

Against this background, we introduce *z-transportability*, the problem of estimating in a *target domain* the causal effect of a set of variables $\mathbf{X}$ on another set of variables $\mathbf{Y}$ from experiments on an arbitrary set of controllable variables $\mathbf{Z}$ in a *source domain.* z-Transportability generalizes *z-identifiability* (Bareinboim and Pearl, 2012a), the problem of estimating in a given domain, the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ from surrogate experiments on $\mathbf{Z}$. z-Transportability also generalizes transportability (Pearl and Bareinboim, 2011) which requires only that the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ in the target domain be estimable from experiments on $\mathbf{V}$ (where $\mathbf{V}$ is the set of *all* variables, including those included in $\mathbf{X}$ and those included in $\mathbf{Z}$) in the source domain. We first generalize z-identifiability to allow cases where $\mathbf{Z}$ is not necessarily disjoint from $\mathbf{X}$. Then, we establish a necessary and sufficient condition for z-transportability and relate it to the corresponding conditions for generalized z-identifiability (Bareinboim and Pearl, 2012a) and for transportability (Bareinboim and Pearl, 2012b). We provide a correct and *complete* algorithm that determines whether a causal effect is *z-transportable*; and if it is, produces a *transport formula*, that is, a recipe for estimating the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ in the target domain using information elicited from the results of experimental manipulations of $\mathbf{Z}$ in the source domain and observational data from the target domain.

This work was carried out independently of Bareinboim and Pearl (2013a).[1] The key differences between (Bareinboim and Pearl, 2013a) and this paper are that (i) we establish a necessary and sufficient condition for z-transportability directly from existing results for generalized z-identifiability and transportability whereas Bareinboim and Pearl (2013a) introduces a graphical criterion called *zs*-hedge. In addition, our algorithm differs from that described in (Bareinboim and Pearl, 2013a) in how it goes about determining whether a causal effect is *z-transportable* (and if it is, computing a transport formula).

[1]Our work was completed in January 2013 and submitted to UAI 2013 on March 1, 2013. We learned of the results in (Bareinboim and Pearl, 2013a) when it appeared as a Technical Report in April, 2013 after its acceptance for publication in AAAI 2013 while our UAI 2013 submission was still under review.

The rest of the paper is organized as follows: Section 2 reviews some of the basic notions, essential definitions, and basic results that set the stage for the rest of the paper; Section 3 generalizes z-identifiability to remove disjoint assumptions on $\mathbf{Z}$; Section 4 introduces z-transportability and establishes a set of necessary and sufficient conditions for z-transportability. Section 5 describes an algorithm for z-transportability and proves its soundness and completeness. Section 6 concludes with a summary and an outline of some promising directions for further research.

2 PRELIMINARIES

Here we introduce some basic notations, review some basic notions, essential definitions, and basic results that set the stage for the rest of the paper.

We adopt notational convention established in the literature on identifiability (Tian and Pearl, 2002; Shpitser and Pearl, 2006b; Bareinboim and Pearl, 2012b,a). Variables are denoted by capital letters, e.g., X, Y and their valuations or realizations by their lowercase counterparts, e.g., x, y. Bold letters e.g., $\mathbf{X}$ are used to denote sets of variables and their values e.g., $\mathbf{x}$. A directed acyclic graph (DAG) is denoted by G and its vertices are denoted by $\mathbf{V}$. The set of *ancestors* of a variable W in a graph G (including W) is denoted by $An\left(W\right)_G$. We use $An\left(\mathbf{W}\right)_G$ to denote $\bigcup_{W\in\mathbf{W}} An\left(W\right)_G$. We denote by $G\left[\mathbf{Y}\right]$, a subgraph of G containing nodes in $\mathbf{Y}$ and all edges between the corresponding nodes in G. Following (Pearl, 2000), we denote by $G_{\overline{\mathbf{X}}}$, the edge subgraph of G where all incoming arrows into nodes in $\mathbf{X}$ are deleted; by $G_{\underline{\mathbf{Y}}}$, the edge subgraph of G where all outgoing arrows from nodes in $\mathbf{Y}$ are deleted; and by $G_{\overline{\mathbf{X}}\underline{\mathbf{Y}}}$, the edge subgraph of G where all incoming arrows into nodes in $\mathbf{X}$ and all outgoing arrows from nodes in $\mathbf{Y}$ are deleted.

We now proceed to review some key definitions and results.

A *causal diagram* (Pearl, 2000) G is a semi-Markovian graph (i.e., a graph with directed as well as bidirected edges that does not have directed cycles) which encodes a set of causal assumptions. A *causal model* (Pearl, 2000) is a tuple $\langle \mathbf{U}, \mathbf{V}, F\rangle$ where $\mathbf{U}$ is a set of *background* or *hidden* variables that cannot be observed or experimented on but which can influence the rest of the model; $\mathbf{V}$ is a set of observed variables $\{V_1, \dots V_n\}$ that are determined by variables in the model, i.e., variables in $\mathbf{U}\cup\mathbf{V}$; F is a set of deterministic functions $\{f_1, \dots, f_n\}$ where each f_i specifies the value of the observed variable V_i given the values of observable parents of V_i and the values of hidden causes of V_i. A *probabilistic causal model* (Pearl, 2000) (PCM) is a tuple $M = \langle \mathbf{U}, \mathbf{V}, F, P\left(\mathbf{U}\right)\rangle$ where $P\left(\mathbf{U}\right)$

is a joint distribution over $\mathbf{U}$.

Intervention (Pearl, 2000) on a set of variables $\mathbf{X} \subseteq \mathbf{V}$ of PCM $M = \langle \mathbf{U}, \mathbf{V}, F, P(\mathbf{U}) \rangle$ involves setting to $\mathbf{X} = \mathbf{x}$ and is denoted by *do*-operation $do(\mathbf{X} = \mathbf{x})$ or simply $do(\mathbf{x})$. A *causal effect* of $\mathbf{X}$ on a disjoint set of variables $\mathbf{Y} \subseteq \mathbf{V} \setminus \mathbf{X}$ is written as $P(\mathbf{y}|do(\mathbf{x}))$ or simply $P_{\mathbf{x}}(\mathbf{y})$. Intervention on a set of variables $\mathbf{X} \subseteq \mathbf{V}$ creates a *submodel* (Pearl, 2000) $M_{\mathbf{x}}$ of M defined as follows: $M_{\mathbf{x}} = \langle \mathbf{U}, \mathbf{V}, F_{\mathbf{x}}, P(\mathbf{U}) \rangle$ where $F_{\mathbf{x}}$ is obtained by taking a set of distinct copies of functions in F and replacing the functions that determine the value of variables in $\mathbf{X}$ by constant functions setting the variables to values $\mathbf{x}$. It is easy to see that a causal diagram G that encodes the causal assumptions of model M is modified to $G_{\overline{\mathbf{X}}}$ by intervention on $\mathbf{X}$.

Definition 1 (Causal Effects Identifiability (Pearl, 2000))**.** Let $\mathbf{X}$, $\mathbf{Y}$ be two sets of disjoint variables, and let G be the causal diagram. The causal effect of an action $do(\mathbf{X} = \mathbf{x})$ on a set of variables $\mathbf{Y}$ is said to be *identifiable* from P in G if $P_{\mathbf{x}}(\mathbf{y})$ is (uniquely) computable from $P(\mathbf{V})$ in any model that induces G.

Do-calculus (Pearl, 1995) offers a sound and complete (Shpitser and Pearl, 2006b; Huang and Valtorta, 2006) inferential machinery for deciding identifiability (Tian and Pearl, 2002; Tian, 2004; Shpitser and Pearl, 2006a) in the sense that, if a causal effect is identifiable, there exists a sequence of applications of the rules of *do*-calculus that transforms the causal effect into a formula that includes only observational quantities. Let G be a causal diagram and P be a distribution on G. Let $\mathbf{W}$, $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{T}$ be disjoint sets of variables in G. Then, the three rules of *do*-calculus are (Pearl, 1995):

(Rule 1) Insertion/deletion of observations:
$P_{\mathbf{x}}(\mathbf{y} \mid \mathbf{t}, \mathbf{w}) = P_{\mathbf{x}}(\mathbf{y} \mid \mathbf{w})$ if $(\mathbf{Y} \perp\!\!\!\perp \mathbf{T} \mid \mathbf{X}, \mathbf{W})_{G_{\overline{\mathbf{X}}}}$

(Rule 2) Intervention/observation exchange:
$P_{\mathbf{x},\mathbf{t}}(\mathbf{y} \mid \mathbf{w}) = P_{\mathbf{x}}(\mathbf{y} \mid \mathbf{t}, \mathbf{w})$ if $(\mathbf{Y} \perp\!\!\!\perp \mathbf{T} \mid \mathbf{X}, \mathbf{W})_{G_{\overline{\mathbf{X}}\underline{\mathbf{T}}}}$

(Rule 3) Insertion/deletion of interventions:
$P_{\mathbf{x},\mathbf{t}}(\mathbf{y} \mid \mathbf{w}) = P_{\mathbf{x}}(\mathbf{y} \mid \mathbf{w})$ if $(\mathbf{Y} \perp\!\!\!\perp \mathbf{T} \mid \mathbf{X}, \mathbf{W})_{G_{\overline{\mathbf{X}}, \overline{\mathbf{T}(\mathbf{W})}}}$ where $\mathbf{T}(\mathbf{W})$ represents $\mathbf{T} \setminus An(\mathbf{W})_{G_{\overline{\mathbf{X}}}}$.

Shpitser and Pearl (2006b) devised an efficient and complete algorithm, **ID**, for identifying causal effects. **ID** employs *c-component decomposition* of a graph and the resulting factorization of a causal effect (Tian and Pearl, 2002) which can be expressed in terms of standard probability manipulations and *do*-calculus.

Definition 2 (*C*-component)**.** Let G be a semi-Markovian graph such that a subset of its bidirected arcs forms a spanning tree over all vertices in G. Then G is a *c-component* (confounded component).

We denote the set of c-components in G by $\mathcal{C}(G)$.

Pearl and Bareinboim (2011) defined *transportability* which offers a license to transport causal information learned from *experimental* studies in a *source* domain to a *target* domain in which only an *observational* study is possible. They also introduced a *selection diagram*, a graphical representation for combining a causal diagram in a source with a causal diagram in a target domain.

Definition 3 (Selection Diagram (Pearl and Bareinboim, 2011))**.** Let $\langle M, M^* \rangle$ be a pair of structural causal models relative to domains $\langle \Pi, \Pi^* \rangle$, sharing a causal diagram G. $\langle M, M^* \rangle$ is said to induce a *selection diagram* D if D is constructed as follows: (i) every edge in G is also an edge in D; (ii) D contains an extra edge $S_i \rightarrow V_i$ whenever there might exist a discrepancy $f_i \neq f_i^*$ or $P(U^i) \neq P^*(U^i)$ between M and M^*.

We call the set of such S_i selection variables and denote them by $\mathbf{S}$.

Definition 4 (Causal Effects Transportability (Pearl and Bareinboim, 2011))**.** Let D be a selection diagram relative to domains $\langle \Pi, \Pi^* \rangle$. Let $\langle P, I \rangle$ be the pair of observational and interventional distributions of Π, and P^* be the observational distribution of Π^*. The causal effect $R = P_{\mathbf{x}}(\mathbf{y})$ is said to be *transportable* from Π to Π^* in D if $P^*_{\mathbf{x}}(\mathbf{y})$ is uniquely computable from P, P^*, I in any model that induces D.

Bareinboim and Pearl (2012b) provided **sID**, an algorithm for transporting causal effects from one domain to another. **sID** is an extension of **ID** (Shpitser and Pearl, 2006b), an algorithm for identifying causal effects from experiments and observations.

Bareinboim and Pearl (2012a) introduced *z-identifiability*, the problem of estimating in a given domain the effect on a set of variables $\mathbf{Y}$ of interventions on a set of variables $\mathbf{X}$ from surrogate experiments on a different set, $\mathbf{Z}$, that is more accessible to manipulation than $\mathbf{X}$.

Definition 5 (Causal Effects *z*-Identifiability (Bareinboim and Pearl, 2012a))**.** Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be disjoint subsets of observable variables $\mathbf{V}$, and let G be the causal diagram. The causal effect of an action $do(\mathbf{X} = \mathbf{x})$ on a set of variables $\mathbf{Y}$ is said to be *z-identifiable* from P in G if $P_{\mathbf{x}}(\mathbf{y})$ is (uniquely) computable from $P(\mathbf{V})$ together with the set of interventional distributions $I_{\mathbf{Z}} = \{P(\mathbf{V} \setminus \mathbf{Z}' \mid do(\mathbf{Z}'))\}_{\mathbf{Z}' \in \mathcal{P}(\mathbf{Z}) \setminus \{\emptyset\}}$, in any model that induces G.

Bareinboim and Pearl (2012a) established a graphical necessary and sufficient condition for *z-identifiability* for arbitrary disjoint sets of variables $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$:

Theorem 1. *The causal effect* $R = P(\mathbf{y} \mid do(\mathbf{x}))$ *is* *zID in G if and only if one of the following conditions hold:*

1. *R is identifiable in G; or*
2. *There exists $\mathbf{Z}' \subseteq \mathbf{Z}$ such that the following conditions hold, (a) $\mathbf{X}$ intercepts all directed paths from $\mathbf{Z}'$ to $\mathbf{Y}$, and (b) R is identifiable in $G_{\overline{\mathbf{Z}'}}$.*

Bareinboim and Pearl (2012a) also established the completeness of *do*-calculus relative to z-identifiability. They also provided $\mathbf{ID^z}$, a complete algorithm for computing the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ using information provided by experiments on $\mathbf{Z}$ under the assumption that $\mathbf{Z} \cap \mathbf{X} = \emptyset$.

3 GENERALIZED z-IDENTIFIABILITY

We proceed to generalize z-identifiability to allow cases where $\mathbf{Z}$ is not necessarily disjoint from $\mathbf{X}$.[2]

Definition 6 (Generalized Causal Effects z-Identifiability)**.** Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be arbitrary subsets of observable variables $\mathbf{V}$ with $\mathbf{X} \cap \mathbf{Y} = \emptyset$, and let G be the causal diagram. The causal effect of an action $do(\mathbf{X} = \mathbf{x})$ on a set of variables $\mathbf{Y}$ is said to be *gz-identifiable* from P in G if $P_{\mathbf{x}}(\mathbf{y})$ is (uniquely) computable from $P(\mathbf{V})$ together with the set of interventional distributions $I_{\mathbf{Z}} = \{P(\mathbf{V} \setminus \mathbf{Z}' \mid do(\mathbf{Z}'))\}_{\mathbf{Z}' \in \mathcal{P}(\mathbf{Z}) \setminus \{\emptyset, \mathbf{V}\}}$, in any model that induces G.

We note that the assumption of $\mathbf{Y}$ and $\mathbf{Z}$ being disjoint can be trivially ignored since experiments on $\mathbf{Y} \cap \mathbf{Z}$ have no bearing on identification of a causal effect on $\mathbf{Y}$. In addition, the assumption in the definition of z-identifiability that $\mathbf{Z}$ and $\mathbf{X}$ are disjoint can be *easily* dropped since identifying $P_{\mathbf{x}}(\mathbf{y})$ in G is *identical* to identifying $P_{\mathbf{x} \setminus \mathbf{z}}(\mathbf{y})$ in $G_{\overline{\mathbf{Z} \cap \mathbf{X}}}$.

The necessary and sufficient conditions for *gzID* can follow immediately from Theorem 1 with minor modifications to allow for the possibility that $\mathbf{Z}$ may not necessarily be disjoint from $\mathbf{X}$:

Theorem 2. *The causal effect $R = P(\mathbf{y} \mid do(\mathbf{x}))$ is gzID in G if and only if one of the following conditions hold:*

1. *R is identifiable in G; or*

[2]We will use the abbreviations *ID*, *zID*, *gzID*, *TR*, and *zTR* respectively to denote identifiability, z-identifiability, gz-identifiability, transportability, and z-transportability, respectively, when used as nouns; and identifiable, z-identifiable, gz-identifiable, transportable, z-transportable, respectively, when used as adjectives.

```
function GID^z $(\mathbf{y}, \mathbf{x}, \mathbf{Z}, \mathcal{I}, \mathcal{J}\, P, G)$
INPUT: $\mathbf{x}$,$\mathbf{y}$: value assignments; $\mathbf{Z}$: variables with interventions available; $\mathcal{I}$, $\mathcal{J}$: active experiments; $P$: current probability distribution $do(\mathcal{I}, \mathcal{J}, \mathbf{x})$ (observational when $\mathcal{I} = \mathcal{J} = \emptyset$); $G$: a causal graph;
OUTPUT: Expression for $P_{\mathbf{x}}(\mathbf{y})$ in terms of $P$, $P_{\mathbf{z}}$ or FAIL$(F, F')$.
1 if $\mathbf{X} = \emptyset$, return $\sum_{\mathbf{v} \setminus \mathbf{y}} P(\mathbf{v})$
2 if $\mathbf{V} \setminus An(\mathbf{Y})_G \neq \emptyset$,
    return GID^z$(\mathbf{y}, \mathbf{x} \cap An(\mathbf{Y})_G, \mathbf{Z},$
                 $\mathcal{I}, \mathcal{J}, \sum_{\mathbf{v} \setminus An(\mathbf{Y})_G} P, An(\mathbf{Y})_G)$
3 Set $\mathbf{W} = \mathbf{V} \setminus (\mathbf{X} \cup \mathcal{I} \cup \mathcal{J})) \setminus An(\mathbf{Y})_{G_{\overline{\mathbf{X} \cup \mathcal{I} \cup \mathcal{J}}}}$
  Set $\mathbf{Z_w} = \mathbf{Z} \cap (\mathbf{X} \cup \mathbf{W})$
  if $(\mathbf{Z_w} \cup \mathbf{W}) \neq \emptyset$,
    return GID^z$(\mathbf{y}, \mathbf{x} \cup \mathbf{w} \setminus \mathbf{Z_w}, \mathbf{Z} \setminus \mathbf{Z_w},$
                 $\mathcal{I} \cup \mathbf{z_w}, \mathcal{J}, P, G)$
4 if $\mathcal{C}(G \setminus (\mathbf{X} \cup \mathcal{I} \cup \mathcal{J})) = \{C_0, \ldots, C_k\}$,
    return $\sum_{\mathbf{v} \setminus \{\mathbf{y}, \mathbf{x}, \mathcal{I}\}} \prod_i$ GID^z$(c_i, (\mathbf{v} \setminus c_i) \setminus \mathbf{Z},$
                 $\mathbf{Z} \setminus (\mathbf{V} \setminus C_i), \mathcal{I}, \mathcal{J} \cup (\mathbf{Z} \cap (\mathbf{v} \setminus c_i)), P, G)$
  if $\mathcal{C}(G \setminus (\mathbf{X} \cup \mathcal{I} \cup \mathcal{J})) = \{C\}$,
5   if $\mathcal{C}(G) = \{G\}$, throw FAIL$(G, C)$
6   if $C \in \mathcal{C}(G)$,
      return $\sum_{c \setminus \mathbf{y}} \prod_{i \mid V_i \in C} P(v_i \mid v_G^{(i-1)} \setminus (\mathcal{I} \cup \mathcal{J}))$
7   if $(\exists C')\, C \subset C' \in \mathcal{C}(G)$,
      return GID^z$(\mathbf{y}, \mathbf{x} \cap C', \mathbf{Z}, \mathcal{I}, \mathcal{J},$
      $\prod_{i \mid V_i \in C'} P(V_i \mid V_G^{(i-1)} \cap C', v_G^{(i-1)} \setminus (C' \cup \mathcal{I} \cup \mathcal{J})), C')$
```

Figure 1: $\mathbf{GID^z}$ for *gzID*.

2. *There exists $\mathbf{Z}' \subseteq \mathbf{Z}$ such that the following conditions hold, (a) $\mathbf{X}$ intercepts all directed path from $\mathbf{Z}' \setminus \mathbf{X}$ to $\mathbf{Y}$, and (b) $P(y \mid do(\mathbf{x} \setminus \mathbf{z}'))$ is identifiable in $G_{\overline{\mathbf{Z}'}}$*

One may simply call $\mathbf{ID^z}$ by passing $P_{\mathbf{x} \setminus \mathbf{z}}(\mathbf{y})$ in $G_{\overline{\mathbf{Z} \cap \mathbf{X}}}$ with surrogate variables $\mathbf{Z} \setminus \mathbf{X}$ (i.e., wrapping $\mathbf{ID^z}$) to yield a sound and complete algorithm for *gzID*. Instead, we obtain $\mathbf{GID^z}$ (Figure 1) by making a minor modification to the $\mathbf{ID^z}$ algorithm (Bareinboim and Pearl, 2012a) (on line 3) which reflects Theorem 2. This only delays the use of experiments on $\mathbf{X} \cap \mathbf{Z}$ to line 3 to allow for the possibility that $\mathbf{Z} \cap \mathbf{X} \neq \emptyset$.

4 z-TRANSPORTABILITY

We introduce *z-transportability*, the problem of estimating the effect on a set of variables $\mathbf{Y}$ of interventions on a set of variables $\mathbf{X}$ in a *target domain* from experiments on an arbitrary set of controllable variables $\mathbf{Z}$ in a *source domain.*

Definition 7 (Causal Effects z-Transportability)**.** Let $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ be sets of variables where $\mathbf{X}$ is disjoint from $\mathbf{Y}$. Let D be a selection diagram relative to domains $\langle \Pi, \Pi^* \rangle$. Let P be the observational distribution and $I_{\mathbf{Z}} = \{P(\mathbf{V} \setminus \mathbf{Z}' \mid do(\mathbf{Z}'))\}_{\mathbf{Z}' \in \mathcal{P}(\mathbf{Z}) \setminus \{\emptyset, \mathbf{V}\}}$ the set of interventional distributions of Π, and P^* the observational distribution of Π^*. The causal effect $P_{\mathbf{x}}(\mathbf{y})$ is said to be *z-transportable* from Π to Π^* in D if $P^*_{\mathbf{x}}(\mathbf{y})$ is uniquely computable from P, P^*, $I_{\mathbf{Z}}$ in any model that induces D.

Thus, z-transportability requires that the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ in a target domain be estimable from experiments on $\mathbf{Z}$ in a source domain when only the variables $\mathbf{Z}$ are controllable and any subset of $\mathbf{Z}$ can be controlled together. It is easy to see that z-transportability generalizes *gzID*, the problem of estimating in a given domain, the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ from experiments on $\mathbf{Z}$. z-Transportability also generalizes *TR* which requires only that the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ in the target domain be estimable from experiments on $\mathbf{V}$ in the source domain.

Lemma 1. *Let $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ be sets of variables with $\mathbf{X}$ disjoint from $\mathbf{Y}$, in population Π and Π^*, and let D be the selection diagram characterizing Π and Π^*. $P(\mathbf{y} \mid do(\mathbf{x}))$ is not z-transportable from Π to Π^* if there exist two causal models M^1 and M^2 compatible with D such that $P^*_1(\mathbf{V}) = P^*_2(\mathbf{V})$, $P_1(\mathbf{V}) = P_2(\mathbf{V})$, $P_1(\mathbf{V} \setminus \mathbf{Z}' \mid do(\mathbf{Z}')) = P_2(\mathbf{V} \setminus \mathbf{Z}' \mid do(\mathbf{Z}'))$, for all $\mathbf{Z}' \subseteq \mathbf{Z}$ and $\mathbf{Z}' \neq \mathbf{V}$, and $P^*_1(\mathbf{y} \mid do(\mathbf{x})) \neq P^*_2(\mathbf{y} \mid do(\mathbf{x}))$.*

Proof. The non-uniqueness of $P^*(\mathbf{y} \mid do(\mathbf{x}))$ implies that there is no function that maps from P, P^*, $I_{\mathbf{Z}}$ to $P^*(\mathbf{y} \mid do(\mathbf{x}))$. □

Lemma 2. *Let $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ be sets of variables with $\mathbf{X}$ disjoint from $\mathbf{Y}$. Let D be a selection diagram characterizing Π and Π^*, and $\mathbf{S}$ be a set of selection variables in D. The causal effect $R = P(\mathbf{y} \mid do(\mathbf{x}))$ is z-transportable from Π to Π^* in D if $P(\mathbf{y} \mid do(\mathbf{x}), \mathbf{s})$ is reducible, using the rules of do-calculus, to an expression in which: $\mathbf{S}$ appears only as a conditioning variable in do-free terms; and interventions in do-terms are a subset of $\mathbf{Z}$.*

Proof. An expression that is a transport formula for $P(\mathbf{y} \mid do(\mathbf{x}), \mathbf{s})$ can contain only P^*, P (terms without the *do*-operator) and $I_{\mathbf{Z}}$ (terms that contain the *do*-operator on a subset of $\mathbf{Z}$ and but not the selection variables). By the correctness of *do*-calculus, the existence of the formula implies z-transportability of $P^*(\mathbf{y} \mid do(\mathbf{x}))$. □

Figure 2 shows selection diagrams where $P_x(y)$ is not *ID* but *zTR* given an experiment on Z. The causal effect $P^*_x(y)$ in each graph is uniquely estimable using the rules of *do*-calculus. By adding experiments on Z by rule 3, we get: $P_x(y \mid s) = P_{z,x}(y \mid s)$. We can eliminate the effect of selection variable (■) on the two domains using rule 1 to obtain: $P_{z,x}(y \mid s) = P_{z,x}(y)$. Except in Figure 2(b), $P_{z,x}(y)$ can be expressed using rule 2 as: $P_{z,x}(y) = P_z(y \mid x)$. In Figure 2(b) we have: $P_{z,x}(y) = \sum_w P_z(w) P_z(y \mid w, x)$.

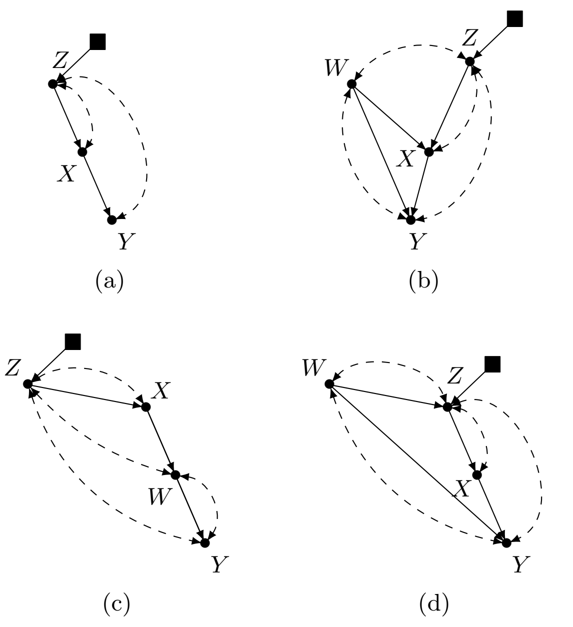


Figure 2: Selection diagrams where $P_x(y)$ is *zTR*, but not *ID*, with an experiment on Z. A selection variable S is represented by a black square ■.

If $P_x(y)$ is *non-gzID* or *non-TR*, then the causal effect is *non-zTR* (see Lemma 3). For example, in the case of the four-node selection diagrams in Figure 2, if a controllable variable is W instead of Z, then $P_x(y)$ is *non-gzID* and hence *non-zTR*. Similarly, if a selection variable is pointing to W instead of Z, then $P_x(y)$ is *non-TR* and hence *non-zTR*.

We now proceed to establish the necessary and sufficient conditions for *zTR*. We start with a lemma that asserts a necessary condition for *zTR* in terms of *TR* and *gzID*.

Lemma 3 (Necessity). *A causal effect $R = P_{\mathbf{x}}(\mathbf{y})$ is z-transportable from Π to Π^* in D only if R is gz-identifiable from P and $I_{\mathbf{Z}}$ in G and R is transportable from Π to Π^* in D.*

Proof. This follows from the definitions of *gzID*, *TR* and *zTR*. First, *gzID* is a special case of *zTR* where $\Pi = \Pi^*$ ($\mathbf{S} = \emptyset$). In addition, *TR* is a special case of *zTR* where $\mathbf{Z} = \mathbf{V}$. Since no difference between two domains, $\mathbf{S} = \emptyset$, or availability of all experiments, $\mathbf{Z} = \mathbf{V}$, make the problem easier, *zTR* of R implies *gzID* of R and *TR* of R. It then follows that general z-identifiability of R and transportability of R are *necessary* for z-transportability of R. □

Since every *zTR* relation satisfies *gzID* and *TR*, we proceed to examine *TR* and *gzID* in depth. Bareinboim and Pearl (2012b) observed that a *TR* causal relation can be decomposed into *trivially transportable* and *directly transportable* (*DTR* for short) relations.

Definition 8 (Trivial Transportability)**.** A causal relation R is said to be *trivially transportable* from Π to Π^*, if $R(\Pi^*)$ is identifiable from (G^*, P^*).

Definition 9 (Direct Transportability)**.** A causal relation R is said to be *directly transportable* from Π to Π^*, if $R(\Pi^*) = R(\Pi)$.

The equality of relations $P^*_{\mathbf{x}}(\mathbf{y})$ and $P_{\mathbf{x}}(\mathbf{y})$ holds if $(\mathbf{S} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{D_{\overline{\mathbf{X}}}}$ by rule 1 of *do*-calculus.

Recall that a causal effect R can be factorized into multiple causal effects (*c-factors*) based on *c-components* (Tian and Pearl, 2002) and *gzID* of R can be determined using a divide-and-conquer strategy. Hence, a causal effect is *gzID* if and only if each *c-factor* resulting from the factorization of R is identifiable by the condition 1 or 2 in Theorem 2. It therefore follows that given a causal relation R that is both *TR* and *gzID* , if one of the *c-factors* of R is not *ID* in the target domain, then that *c-factor* must be *DTR* from the source domain and *ID* from $P_{\mathbf{Z}'}$ in an edge subgraph $G_{\overline{\mathbf{Z}'}}$ of a causal diagram G (condition 2 in Theorem 2). We provide a basic lemma for the factorization of a causal effect based on (Tian and Pearl, 2002; Shpitser and Pearl, 2006b).

Lemma 4. *Let* $\mathbf{X}$ *and* $\mathbf{Y}$ *be disjoint sets of variables of* G*. Let* $G' = G[An(\mathbf{Y})_G]$, $\mathbf{X}' = \mathbf{X} \cap An(\mathbf{Y})_G$, *and* $\mathbf{V}'$ *be variables in* G' *and* $\mathbf{v}'$ *their valuations. Let* $\mathbf{W} = \mathbf{V}' \setminus \mathbf{X}' \setminus An(\mathbf{Y})_{G'_{\overline{\mathbf{X}'}}}$, $\mathbf{X}'' = \mathbf{X}' \cup \mathbf{W}$*;* $\mathcal{C}(G' \setminus \mathbf{X}'')$ *a c-component decomposition of graph* $G' \setminus \mathbf{X}''$*; and* $\mathbf{Q} = \{P_{\mathbf{v}' \setminus c_i}(c_i)\}_{C_i \in \mathcal{C}(G' \setminus \mathbf{X}'')}$ *the set of corresponding c-factors. Then* $P_{\mathbf{x}}(\mathbf{y}) = \sum_{\mathbf{v}' \setminus \{\mathbf{x}'' \cup \mathbf{y}\}} \prod_{Q_i \in \mathbf{Q}} Q_i$.

Proof. The proof follows from Lemma 3 (Shpitser and Pearl, 2006b). □

Lemma 5. *Let* G *be a common causal diagram of domains* Π *and* Π^**. Let* $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ *be sets of variables of* G *with* $\mathbf{X}$ *disjoint from* $\mathbf{Y}$ *and* $\mathbf{v}$ *a valuation of* $\mathbf{V}$*. Let* $G' = G[An(\mathbf{Y})_G]$, $\mathbf{X}' = \mathbf{X} \cap An(\mathbf{Y})_G$, *and* $\mathbf{V}'$ *be variables in* G' *and* $\mathbf{v}'$ *their valuations. Let* $\mathbf{W} = \mathbf{V}' \setminus \mathbf{X}' \setminus An(\mathbf{Y})_{G'_{\overline{\mathbf{X}'}}}$, $\mathbf{X}'' = \mathbf{X}' \cup \mathbf{W}$*;* $\mathcal{C}(G' \setminus \mathbf{X}'')$ *a c-component decomposition of graph* $G' \setminus \mathbf{X}''$*; and* $\mathbf{Q} = \{P_{\mathbf{v}' \setminus c_i}(c_i)\}_{C_i \in \mathcal{C}(G' \setminus \mathbf{X}'')}$ *the set of corresponding c-factors. If* $R = P_{\mathbf{x}}(\mathbf{y})$ *is gzID from* Π *and TR from* Π *to* Π^**, then for every* $Q_i \in \mathbf{Q}$ *the following conditions hold:*

(i) Q_i *is identifiable from* $P^*(\mathbf{V}')$ *in* G'*; or*

(ii) Q_i *is (a) identifiable from* $P_{\mathbf{z}'}$ *in* $G'_{\overline{\mathbf{Z}'}}$ *where* $\mathbf{Z}' = (\mathbf{V}' \setminus C_i) \cap \mathbf{Z}$ *and (b) directly transportable from* Π *to* Π^*.

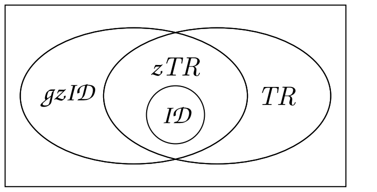


Figure 3: A Venn diagram depicting Theorem 3. Given a selection diagram D and controllable variables $\mathbf{Z}$, the intersection of *gzID* and *TR* relations exactly matches to *zTR* relations in any model that induces D.

Proof. Let $\mathbf{Q}_1$ be a set of *all ID* causal effects in $\mathbf{Q}$ and let $\mathbf{Q}_2$ be $\mathbf{Q} \setminus \mathbf{Q}_1$. If each causal effect in $\mathbf{Q}_2$ does not satisfy the second condition, it contradicts the premise that R is *gzID* and *TR*. □

Lemma 6 (Sufficiency)**.** *Let* $\mathbf{X}$, $\mathbf{Y}$, *and* $\mathbf{Z}$ *be sets of variables of* G *with* $\mathbf{X}$ *disjoint from* $\mathbf{Y}$*. If* $R = P_{\mathbf{x}}(\mathbf{y})$ *is gzID in* Π *and TR from* Π *to* Π^**, then* R *is zTR from* Π *to* Π^*.

Proof. By Lemma 5, for every *c-factor* $Q \in \mathbf{Q}$ that is not *ID* in Π^*, there exists a subset $\mathbf{Z}$ of $\mathbf{V}$ such that Q is identifiable in Π using experiments on $\mathbf{Z}$ and *DTR* of Q. Hence, the causal effect R is reducible to an expression where every term from *c-factors* in $\mathbf{Q}$ is either (i) a *do*-free term (i.e., identified from P^*) or (ii) a term that contains the *do*-operator but no selection variables (i.e., identified from $I_{\mathbf{Z}}$) in which the intervention is on a subset of $\mathbf{Z}$. Therefore, R is *zTR* from Π to Π^*. □

The necessary and sufficient conditions for *zTR* follow from (Lemma 3) and sufficiency (Lemma 6).

Theorem 3 (Necessity and Sufficiency)**.** *A causal effect* $R = P_{\mathbf{x}}(\mathbf{y})$ *is zTR from* Π *to* Π^* *in* D *if and only if* (i) R *is gzID from* P *and* $I_{\mathbf{Z}}$ *in* G *and* (ii) R *is TR from* Π *to* Π^* *in* D.

Though Theorem 3 is the main theorem of the paper, it does not directly provide an effective procedure for estimating a causal effect given P^*, P, and $I_{\mathbf{Z}}$. Rather, Lemma 5 will be more instrumental in the design of a complete algorithm for *zTR*.

```
function sIDᶻ (y, x, G, Z)
Input: y, x value assignments for a causal effect $P^*_{\mathbf{x}}(\mathbf{y})$; $G$ a causal diagram; $\mathbf{Z}$ a set of (inactive) controllable variables;
Output: an expression for $P^*_{\mathbf{x}}(\mathbf{y})$ regarding $P^*$ and $I_{\mathbf{Z}}$ or a (s-)hedge.
1 if $\mathbf{X} = \emptyset$, return $P^*(\mathbf{y})$
2 if $\mathbf{V} \neq An(\mathbf{Y})_G$, return $\mathbf{sID^z}(\mathbf{y}, \mathbf{x} \cap An(\mathbf{Y})_G, G[An(\mathbf{Y})_G], \mathbf{Z})$
3 $\mathbf{W} \leftarrow \mathbf{V} \setminus \mathbf{X} \setminus An(\mathbf{Y})_{G_{\overline{\mathbf{X}}}}$
4 if $\mathbf{W} \neq \emptyset$, return $\mathbf{sID^z}(\mathbf{y}, \mathbf{x} \cup \mathbf{w}, G, \mathbf{Z})$
5 $\{C_1, \ldots, C_k\} \leftarrow \mathcal{C}(G \setminus \mathbf{X})$
6 $T_i \leftarrow \{V_j\}_{S_j \in \mathbf{S}} \cap C_i = \emptyset$ for each $i \in \{1, \ldots, k\}$
7 return $\sum_{\mathbf{v}\setminus\{\mathbf{y},\mathbf{x}\}} \prod_{i=1}^{k} T_i ?\, \mathbf{BI}(c_i, \mathbf{v} \setminus c_i \setminus \mathbf{Z}, P_{\mathbf{z}\cap(\mathbf{v}\setminus c_i)}(\mathbf{V}), G_{\overline{\mathbf{Z}\cap(\mathbf{V}\setminus C_i)}}, \mathbf{Z} \cap (\mathbf{V} \setminus C_i)) : \mathbf{BI}(c_i, \mathbf{v} \setminus c_i, P^*(\mathbf{V}), G)$

function BI (y, x, P, G, $\mathcal{I} = \emptyset$)
Input: $P$: current distribution; $G$: current causal diagram; $\mathcal{I}$: active experiments with default value $\emptyset$.
Output: Expression for $P_{\mathbf{x}}(\mathbf{y})$ in terms of $P$ or throw a (s-)hedge depending on the existence of selection variables.
1 if $\mathbf{X} = \emptyset$, return $P(\mathbf{y})$
2 if $\mathbf{V} \neq An(\mathbf{Y})_G$, return $\mathbf{BI}(\mathbf{y}, \mathbf{x} \cap An(\mathbf{Y})_G, P(An(\mathbf{Y})_G), G[An(\mathbf{Y})_G], \mathcal{I} \cap An(\mathbf{Y})_G)$
3 $\{C\} \leftarrow \mathcal{C}(G \setminus (\mathbf{X} \cup \mathcal{I}))$
4 if $\mathcal{C}(G) = \{G\}$, throw FAIL$\langle D[G], D[C]\rangle$
5 if $C \in \mathcal{C}(G)$, return $\sum_{c\setminus\mathbf{y}} \prod_{i|V_i\in C} P(v_i \mid v_G^{(i-1)} \setminus \mathcal{I})$
6 if $(\exists C')\, C \subset C' \in \mathcal{C}(G)$, return $\mathbf{BI}(\mathbf{y}, \mathbf{x} \cap C', \prod_{i|V_i\in C'} P(V_i \mid V_G^{(i-1)} \cap C', v_G^{(i-1)} \setminus (C' \cup \mathcal{I})), C', \mathcal{I} \cap C')$
```

Figure 4: An algorithm for *zTR* with a subroutine **BI** for the identification of a *c*-factor given a (interventional) probability distribution and a (mutilated) graph. To estimate a causal effect $P^*_{\mathbf{x}}(\mathbf{y})$, call $\mathbf{sID^z}(\mathbf{y}, \mathbf{x}, G, \mathbf{Z})$. We assume that P^*, $I_{\mathbf{Z}}$, and D are globally defined for convenience. By construction, $G = G_{\overline{\mathcal{I}}}$ in **BI**. Distribution $P_{\mathbf{z}\cap(\mathbf{v}\setminus c_i)}$ is obtained from $I_{\mathbf{Z}}$.

5 AN ALGORITHM FOR *z*-TRANSPORTABILITY

We proceed to describe $\mathbf{sID^z}$, an algorithm that determines whether a causal relation $P_{\mathbf{x}}(\mathbf{y})$ is *zTR* from Π to Π^*, and if so, produces a correct transport formula; if not, provides an evidence of *non-zTR* (i.e., an *s-hedge* (Bareinboim and Pearl, 2012b) or a *hedge* (Shpitser and Pearl, 2006b; Bareinboim and Pearl, 2012a) if the relation is not *TR* or not *gzID*, respectively).

The design of $\mathbf{sID^z}$ (see Figure 4[3,4]) follows directly from Lemma 5. Specifically, $\mathbf{sID^z}$ factorizes a causal effect based on the decomposition of the given graph into a set of *c*-components. Unlike $\mathbf{GID^z}$ (line 3, Figure 1), $\mathbf{sID^z}$ (Figure 4) postpones covering interventions on $\mathbf{X}$ by experiments on $\mathbf{Z}$ until after the factorization of causal effect until it determines (line 7) whether each *c*-factor can in fact be identified from either the source domain or the target domain. From Lemma 5, each *c*-factor of a *z*-transportable causal effect should be a) identifiable (trivially transportable) in the target domain or b) *gz*-identifiable in the source domain and directly transportable from the source domain to the target domain. Fortunately, direct transportability of a *c*-factor Q_i can be easily computed at the stage of decomposition:

$$\{V_j\}_{S_j\in\mathbf{S}} \cap C_i = \emptyset \tag{1}$$

which is identical to testing S-admissibility (Pearl and Bareinboim, 2011)

$$(\mathbf{S} \perp\!\!\!\perp C_i \mid \mathbf{V} \setminus C_i)_{G_{\overline{\mathbf{V}\setminus C_i}}}.$$

From Theorem 3 above, to establish that a *c*-factor is not *zTR*, it suffices to show the existence of either 1) a hedge (which shows that the causal effect is non *gzID*); or 2) an s-hedge (which shows that the causal effect is non-*TR*). Algorithm $\mathbf{sID^z}$ calls the subroutine **BI** to determine if a directly transportable *c*-factor is *gz*-identifiable. Because ordinarily identifiable *c*-factor is *gz*-identifiable, line 7 of $\mathbf{sID^z}$ employs an inline conditional operator[5]. The subroutine **BI** estimates a *c*-factor given a distribution, a causal graph, and active experiments $\mathcal{I}$. Thus, the algorithm differs from $\mathbf{TR^z}$ (Bareinboim and Pearl, 2013a) which tries to estimate a *c*-factor given an interventional distribution of a source domain *after* the test for trivial transportability of a *c*-factor fails.

The ability to check for direct transportability of a *c*-factor (Equation 1) allows **BI**, upon failure to iden-

[3]For simplicity, we combine $\mathcal{I}$ and $\mathcal{J}$ (see $\mathbf{GID^z}$, Figure 1) into $\mathcal{I}$ (see $\mathbf{sID^z}$, Figure 4)

[4]A manipulated graph $G_{\overline{\mathcal{I}}}$ and experimental distribution $P_{\mathcal{I}}$ are passed as arguments when *active* experiments are set to $\mathcal{I}$. For the use of $G_{\overline{\mathcal{I}}}$ ($G_{\overline{\mathcal{I}\cup\mathcal{J}}}$ in $\mathbf{GID^z}$) we must remove the incoming edges on the active experiments so that a relation can be identified from $P_{\mathbf{Z}'}$ in $G_{\overline{\mathbf{Z}'}}$.

[5]An inline conditional operator is of the form *cond*?*exp*$_1$: *exp*$_2$. The first expression *exp*$_1$ is executed if *cond* is true. Otherwise *exp*$_2$ is executed.

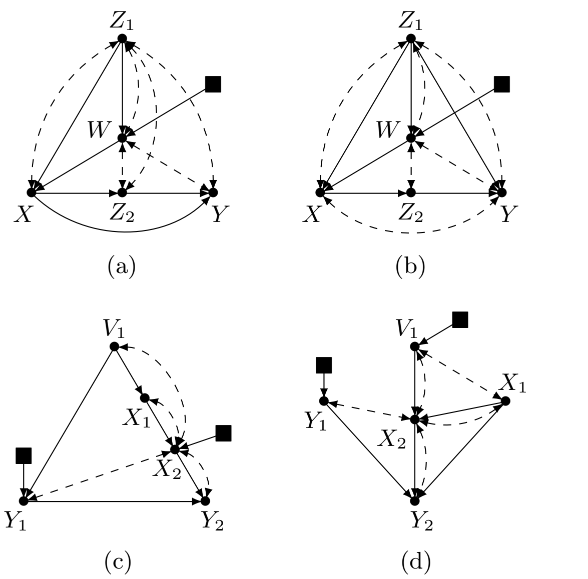


Figure 5: Selection diagrams where $P_{\mathbf{x}}(\mathbf{y})$ is not ID but zTR but given controllable variables $\mathbf{Z}$ (V_1 and X_2 are controllable variables for Figure 5(c) and 5(d)).

tify a c-factor in the source domain, to throw a hedge which implies non-gzID. Similarly, if a c-factor is neither directly-transportable from the source domain to the target domain nor ordinarily identifiable in the target domain, **BI** throws an s-hedge since $\{V_j\}_{S_j\in\mathbf{S}}$ intersects C_i.

5.1 EXAMPLES

Some examples are given in Figure 5 to illustrate how $\mathbf{sID^z}$ estimates a causal effect $P^*_{\mathbf{x}}(\mathbf{y})$ from experiments on $\mathbf{Z}$ from a source domain. We will use $P_{(\mathcal{I}),\mathbf{x}\setminus\mathcal{I}}(\mathbf{y})$ to denote a causal effect $P_{\mathbf{x}}(\mathbf{y})$ in G from an interventional distribution $P_{\mathcal{I}}$ in $G_{\overline{\mathcal{I}}}$ (such that $\mathcal{I}\subseteq\mathbf{X}$).

In Figure 5(a), W and Z_1 are added as interventions (line 4), $P^*_x(y) = P^*_{z_1,w,x}(y)$. Z_1, which is included in controllable variables $\mathbf{Z}$, will not be treated as an active experiment until after the decomposition. The causal effect is factorized as $\sum_{z_2} P^*_{\mathbf{z},w,x}(y)\,P^*_{z_1,w,x,y}(z_2)$. Since no selection variable is pointing to Y or Z_2 (line 6 and 7), the two c-factors are then identified from $P_{\mathbf{z}}$ in $G_{\overline{\mathbf{Z}}}$ and P_{z_1} in $G_{\overline{Z_1}}$, respectively, as

$$\sum_{z_2} P_{(z_1,z_2),w,x}(y)\,P_{(z_1),w,x,y}(z_2)\,.$$

The parameters for the subroutine is $\mathbf{BI}(y,\, w\cup x,\, P_{\mathbf{z}},\, G_{\overline{\mathbf{z}}},\, \mathbf{z})$ and $\mathbf{BI}\left(z_2,\, w\cup x\cup y,\, P_{z_1},\, G_{\overline{Z_1}},\, z_1\right)$, respectively.

In Figure 5(b), $P^*_x(y) = P^*_{w,x}(y)$ by line 4. In lines 5–7, two c-factors $P^*_{z_2,w,x}(z_1,y)$ and $P^*_{z_1,w,x,y}(z_2)$ will be identified in the source domain as

$$\sum_{\mathbf{z}} P_{(z_2),w,x}(z_1,y)\,P_{(z_1),w,x,y}(z_2)\,.$$

In Figure 5(c), $P^*_{\mathbf{x}}(\mathbf{y}) = P^*_{x_1,x_2}(y_1,y_2)$ is factorized as $\sum_{v_1} P^*_{\mathbf{x},\mathbf{y}}(v_1)\,P^*_{v_1,\mathbf{x},y_2}(y_1)\,P^*_{v_1,\mathbf{x},y_1}(y_2)$. Since a selection variable is pointing to Y_1,

$$P^*_{v_1,\mathbf{x},y_2}(y_1) \neq P_{v_1,\mathbf{x},y_2}(y_1) = P_{(v_1,x_2),x_1,y_2}(y_1)\,.$$

Then, the first and third c-factors will be identified in the source domain as

$$\sum_{v_1} P_{(x_2),x_1,\mathbf{y}}(v_1)\,P^*_{v_1,\mathbf{x},y_2}(y_1)\,P_{(v_1,x_2),x_1,y_1}(y_2)\,.$$

In Figure 5(d), V_1 is added to the causal effect as an intervention, $P^*_{\mathbf{x}}(\mathbf{y}) = P^*_{v_1,\mathbf{x}}(\mathbf{y})$ by line 4. By Lemma 4, $P^*_{v_1,\mathbf{x}}(\mathbf{y}) = P^*_{v_1,\mathbf{x},y_2}(y_1)\,P^*_{v_1,\mathbf{x},y_1}(y_2) = P^*_{v_1,\mathbf{x},y_2}(y_1)\,P_{(v_1,x_2),x_1,y_1}(y_2)$.

5.2 SOUNDNESS AND COMPLETENESS

We first illustrate a certain behavior of $\mathbf{GID^z}$ which will help understanding correctness of $\mathbf{sID^z}$.

Remark 1. Addition of interventions (line 3) and decomposition (line 4) of $\mathbf{GID^z}$ are executed at most once.

Proof. Given a causal relation, $\mathbf{GID^z}$ checks whether interventions can be added to the relation (line 3). If so, it adds interventions and the subsequent call checks whether there are multiple c-components in $G\setminus(\mathbf{X}\cup\mathcal{I}\cup\mathcal{J})$ (line 4). If so, the relation is factorized to c-factors; or the relation is already a c-factor. During the estimation of the c-factor, no interventions are added as $(\mathbf{X}\cup(\mathcal{I}\cup\mathcal{J}))\cup\mathbf{Y}=\mathbf{V}$. In addition, decomposition is not required as $G[\mathbf{Y}]$ is a c-component. $\square$

Lemma 7. *Decomposition of* $P_{\mathbf{x}}(\mathbf{y})$ *produced by* $\mathbf{sID^z}$ *is equivalent to that produced by* **sID** *and by* $\mathbf{GID^z}$*.*

Proof. Trivially, the decomposition by **sID** and by $\mathbf{sID^z}$ are identical as **sID** and $\mathbf{sID^z}$ share the same code. As in Remark 1, lines 3 and 4 of $\mathbf{GID^z}$ are executed only once. Hence, when $\mathbf{GID^z}$ adds interventions, $\mathcal{I}$ and $\mathcal{J}$ are empty (line 3) and when it decomposes the relation, $\mathcal{J}$ is empty (line 4). Therefore, $\mathbf{X}\cup\mathbf{W}$ computed by $\mathbf{GID^z}$ (line 3) and by $\mathbf{sID^z}$ (line 4) are identical. Since $\mathcal{I}$ is set to $\mathbf{Z_w}\subseteq\mathbf{X}\cup\mathbf{W}$ by $\mathbf{GID^z}$, $\mathbf{X}\cup\mathcal{I}$ (line 4 of $\mathbf{GID^z}$) is identical to $\mathbf{X}$ in $\mathbf{sID^z}$ (line 5). As a result, $\mathcal{C}(G\setminus(\mathbf{X}\cup\mathcal{I}\cup\mathcal{J}))$ in line 4 of $\mathbf{GID^z}$ is equivalent to $\mathcal{C}(G\setminus\mathbf{X})$ in line 5 of $\mathbf{sID^z}$. $\square$

Lemma 8. *The c-factor estimate produced by* **BI** *are equivalent to those produced by* $\mathbf{GID^z}$*.*

Proof. The code used by **BI** to estimate a c-factor is identical to that used by $\mathbf{GID^z}$ except for lines 3 and 4 of $\mathbf{GID^z}$ (see Footnote 3 and 4). By Remark 1 and Lemma 7, the c-factor estimate produced by **BI** is identical to that produced by $\mathbf{GID^z}$. ☐

Theorem 4 (Soundness)**.** *Whenever* $\mathbf{sID^z}$ *returns an expression for a causal effect* $P^*_{\mathbf{x}}(\mathbf{y})$*, it is correct.*

Proof. The proof follows from Lemma 4 and 5, $\mathbf{sID^z}$ decomposes a causal effect $P^*_{\mathbf{x}}(\mathbf{y})$ and estimates its c-factors either from a source domain or from a target domain. If a c-factor is directly transportable, $\mathbf{sID^z}$ uses experimental distributions from the source domain to estimate it and the call to **BI** yields a c-factor estimate that is identical to that of $\mathbf{GID^z}$ (Lemma 8). If a c-factor is not directly transportable (Equation 1), the c-factor must be trivially identifiable in the target domain and since there are no active experiments, the c-factor estimate produced by **BI** is identical to that produced by **ID** (Lemma 8 with $\mathcal{I} = \emptyset$). ☐

Theorem 5. *Assume* $\mathbf{sID^z}$ *fails to z-transport* $P_{\mathbf{x}}(\mathbf{y})$ *from* Π *to* Π^**. Then* $P_{\mathbf{x}}(\mathbf{y})$ *is neither gz-identifiable from P in G nor transportable from* Π *to* Π^* *in D.*

Proof. If $\mathbf{sID^z}$ fails to z-transport a relation $P_{\mathbf{x}}(\mathbf{y})$ (in line 4 of the subroutine **BI**), it throws a hedge or an s-hedge. The failure is due to the existence of a hedge or an s-hedge associated with a c-factor, say Q. From Lemma 8 it follows that the failure of $\mathbf{sID^z}$ in the case of *empty* active experiments ($\mathcal{I} = \emptyset$) or *nonempty* active experiments ($\mathcal{I} \neq \emptyset$) respectively implies and *non-ID* or *non-gzID* of Q. From the test of direct transportability in line 6 of $\mathbf{sID^z}$ (Equation 1), *non-ID* of Q implies non-direct-transportability of Q. This also implies that $P_{\mathbf{x}}(\mathbf{y})$ is not transportable since there exists a c-factor Q that is neither trivially transportable (*ID*) nor direct-transportable. Therefore, whenever the algorithm fails, $P_{\mathbf{x}}(\mathbf{y})$ is neither *gzID* nor *TR*. ☐

Corollary 1 (Completeness)**.** $\mathbf{sID^z}$ *is complete.*

Proof. This follows from necessity and sufficiency theorem (Theorem 3) and Theorem 5. ☐

The completeness of $\mathbf{sID^z}$ proves that *do*-calculus and standard probability manipulations are sufficient for determining whether a causal effect is z-transportable.

6 SUMMARY AND DISCUSSION

We have introduced z-transportability, the problem of estimating in a target domain the causal effect of a set of variables $\mathbf{X}$ on another set of variables $\mathbf{Y}$ (such that $\mathbf{Y} \cap \mathbf{X} = \emptyset$) from experiments on any subset of an *arbitrary* controllable variables $\mathbf{Z}$ (such that $\mathbf{Z} \subseteq \mathbf{V}$) in a source domain. z-Transportability generalizes z-identifiability (Bareinboim and Pearl, 2012a), the problem of estimating in a given domain the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ from surrogate experiments on $\mathbf{Z}$. z-Transportability also generalizes transportability (Pearl and Bareinboim, 2011) which requires only that the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ in the target domain be estimable from experiments on *all* variables in the source domain. We have generalized z-identifiability to allow cases where $\mathbf{Z}$ is not necessarily disjoint from $\mathbf{X}$. We have established a necessary and sufficient condition for z-transportability in terms of generalized z-identifiability and transportability. We have provided $\mathbf{sID^z}$, an algorithm that determines whether a causal effect is *z-transportable*; and if it is, produces a transport formula, that is, a recipe for estimating the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ in the target domain using information elicited from the results of experimental manipulations of $\mathbf{Z}$ in the source domain and observational data from the target domain. Our results also show that *do*-calculus is complete for z-transportability.

Causal effects identifiability (Galles and Pearl, 1995; Tian, 2004; Tian and Pearl, 2002; Shpitser and Pearl, 2006a,b), transportability (Pearl and Bareinboim, 2011; Bareinboim and Pearl, 2012b), z-identifiability (Bareinboim and Pearl, 2012a), meta-transportability (Bareinboim and Pearl, 2013b; Lee and Honavar, 2013) and z-transportability (introduced in this paper and in Bareinboim and Pearl, 2013a) are all special cases of *meta-identifiability* (Pearl, 2012) which has to do with nonparametric identification of causal effects given *multiple* domains and *arbitrary* information from each domain. Our results suggest several additional special cases of meta-identifiability to consider, including in particular: a generalization of z-transportability that allows causal information from possibly different experiments in *multiple* source domains to be combined to facilitate the estimation of a causal effect in a target domain; variants of z-identifiability that incorporate constraints on simultaneous controllability of combinations of variables; and combinations thereof.

Acknowledgments

The authors are grateful to UAI 2013 anonymous reviewers for their thorough reviews. The work of Vasant Honavar while working at the National Science Foundation was supported by the National Science Foundation. Any opinion, finding, and conclusions contained in this article are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Bareinboim, E., and Pearl, J. 2012a. Causal inference by surrogate experiments: z-identifiability. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, 113–120. AUAI Press.

Bareinboim, E., and Pearl, J. 2012b. Transportability of causal effects: Completeness results. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 698–704. AAAI Press.

Bareinboim, E., and Pearl, J. 2013a. Causal transportability of limited experiments. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*. AAAI Press. To appear.

Bareinboim, E., and Pearl, J. 2013b. Meta-transportability of causal effects: A formal approach. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*. To appear.

Galles, D., and Pearl, J. 1995. Testing identifiability of causal effects. In *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence*, 185–195. Morgan Kaufmann.

Huang, Y., and Valtorta, M. 2006. Pearl's calculus of intervention is complete. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, 217–224. AUAI Press.

Lee, S., and Honavar, V. 2013. m-transportability: Transportability of a causal effect from multiple environments. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*. AAAI Press. To appear.

Pearl, J., and Bareinboim, E. 2011. Transportability of causal and statistical relations: A formal approach. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 247–254. AAAI Press.

Pearl, J. 1995. Causal diagrams for empirical research. *Biometrika* 82(4):669–688.

Pearl, J. 2000. *Causality: models, reasoning, and inference*. New York, NY, USA: Cambridge University Press.

Pearl, J. 2012. The do-calculus revisited. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, 4–11. AUAI Press.

Shpitser, I., and Pearl, J. 2006a. Identification of conditional interventional distributions. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, 437–444. AUAI Press.

Shpitser, I., and Pearl, J. 2006b. Identification of joint interventional distributions in recursive semi-markovian causal models. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 1219–1226. AAAI Press.

Tian, J., and Pearl, J. 2002. A general identification condition for causal effects. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, 567–573. AAAI Press / The MIT Press.

Tian, J. 2004. Identifying conditional causal effects. In *Proceedings of the Twentieth Conference in Uncertainty in Artificial Intelligence*, 561–568. AUAI Press.

A Sound and Complete Algorithm for Learning Causal Models from Relational Data

Marc Maier **Katerina Marazopoulou** **David Arbour** **David Jensen**
School of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003
{maier, kmarazo, darbour, jensen}@cs.umass.edu

Abstract

The PC algorithm learns maximally oriented causal Bayesian networks. However, there is no equivalent complete algorithm for learning the structure of relational models, a more expressive generalization of Bayesian networks. Recent developments in the theory and representation of relational models support lifted reasoning about conditional independence. This enables a powerful constraint for orienting bivariate dependencies and forms the basis of a new algorithm for learning structure. We present the *relational causal discovery* (RCD) algorithm that learns causal relational models. We prove that RCD is sound and complete, and we present empirical results that demonstrate effectiveness.

1 INTRODUCTION

Research in causal discovery has led to the identification of fundamental principles and methods for causal inference, including a *complete* algorithm—the PC algorithm—that identifies all possible orientations of causal dependencies from observed conditional independencies (Pearl, 2000; Spirtes et al., 2000; Meek, 1995). Completeness guarantees that no other method can infer more causal dependencies from observational data. However, much of this work, including the completeness result, applies only to Bayesian networks.

Over the past 15 years, researchers have developed more expressive classes of models, including probabilistic relational models (Getoor et al., 2007), that remove the assumption of independent and identically distributed instances required by Bayesian networks. These *relational* models represent systems involving multiple types of interacting entities with probabilistic dependencies among them. Most algorithms for learning the structure of relational models focus on statistical association. The single algorithm that does address causality—Relational PC (Maier et al., 2010)—is not complete and is prone to orientation errors, as we show in this paper. Consequently, there is no relational analog to the completeness result for Bayesian networks.

Recent advances in the theory and representation of relational models provide a foundation for reasoning about causal dependencies (Maier et al., 2013). That work develops a novel, lifted representation—the *abstract ground graph*—that abstracts over all instantiations of a relational model, and it uses this abstraction to develop the theory of relational d-separation. This theory connects the causal structure of a relational model and probability distributions, similar to how d-separation connects the structure of Bayesian networks and probability distributions.

We present the implications of abstract ground graphs and relational d-separation for learning causal models from relational data. We introduce a powerful constraint that can orient bivariate dependencies (yielding models with up to 72% additional oriented dependencies) without assumptions on the underlying distribution. We prove that this new rule, called *relational bivariate orientation*, combined with relational extensions to the rules utilized by the PC algorithm, yields a sound and complete approach to identifying the causal structure of relational models. We develop a new algorithm, called *relational causal discovery* (RCD), that leverages these constraints, and we prove that RCD is sound and complete under the causal sufficiency assumption. We show RCD's effectiveness with a practical implementation and compare it to several alternative algorithms. Finally, we demonstrate RCD on a real-world dataset drawn from the movie industry.

2 EXAMPLE

Consider a data set containing actors with a measurement of their popularity (e.g., price on the Hollywood Stock Exchange) and the movies they star in with a measurement of success (e.g., box office revenue). A simple analysis might detect a statistical association between popularity and success, but the models in which popularity causes success and success causes popularity may be statistically indistinguishable.

However, with knowledge of the *relational structure*, a considerable amount of information remains to be leveraged. From the perspective of actors, we can ask whether one actor's popularity is conditionally independent of the popularity of other actors appearing in the same movie, given that movie's success. Similarly, from the perspective of movies, we can ask whether the success of a movie is conditionally independent of the success of other movies with a common actor, given that actor's popularity. With conditional independence, we now can determine the orientation for a single *relational dependency*.

These additional tests of conditional independence manifest when inspecting relational data with *abstract ground graphs*—a lifted representation developed by Maier et al. (2013) (see Section 3.2 for more details). If actor popularity indeed causes movie success, then the popularity of actors appearing in the same movie would be marginally independent. This produces a collider from the actor perspective and a common cause from the movie perspective, as shown in Figure 1. With this representation, it is straightforward to identify the orientation of such a bivariate dependency.

This example illustrates two central ideas of this paper. First, abstract ground graphs enable a new constraint on the space of causal models—relational bivariate orientation. The rules used by the PC algorithm can also be adapted to orient the edges of abstract ground graphs (Section 4). Second, this constraint-based approach—testing for conditional independencies and reasoning about them to orient causal dependencies—is the primary strategy of the relational causal discovery algorithm (Section 5).

3 BACKGROUND

The details of RCD and its correctness rely on fundamental concepts of relational data, models, and *d*-separation as provided by Maier et al. (2013). This section provides a review of this theory in the context of the movie domain example. Note that the relational representation is a strictly more general framework for causal discovery, reducing to Bayesian networks in the presence of a single entity with no relationships.

3.1 RELATIONAL DATA AND MODELS

A *relational schema*, $\mathcal{S} = (\mathcal{E}, \mathcal{R}, \mathcal{A})$, describes the entity, relationship, and attribute classes in a domain, as

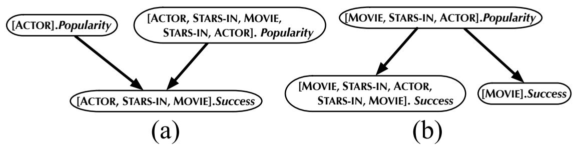
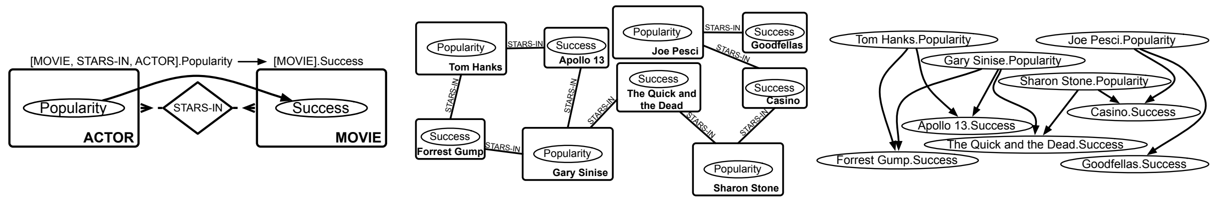

[ACTOR].*Popularity*
[ACTOR, STARS-IN, MOVIE, STARS-IN, ACTOR]. *Popularity*
[ACTOR, STARS-IN, MOVIE].*Success*
(a)
[MOVIE, STARS-IN, ACTOR].*Popularity*
[MOVIE, STARS-IN, ACTOR, STARS-IN, MOVIE]. *Success*
[MOVIE].*Success*
(b)

Figure 1: Abstract ground graphs from (a) the Actor perspective and (b) the Movie perspective.

well as cardinality constraints for the number of entity instances involved in a relationship. A schema is typically depicted with an entity-relationship diagram, such as the one underlying the model shown in Figure 2(a). This example has two entity classes—Actor with attribute *Popularity* and Movie with attribute *Success*—and one relationship class—Stars-In with no attributes. The cardinality constraints (expressed as crow's feet in the diagram) indicate that many actors may appear in a movie and a single actor may appear in many movies. A schema is a template for a *relational skeleton* σ—a data set of entity and relationship instances. The example in Figure 2(b) contains four Actor instances, five Movie instances, and the relationships among them.

Given a relational schema, one can specify *relational paths*, which are critical for specifying the variables and dependencies of a relational model. A relational path is an alternating sequence of entity and relationship classes that follow connected paths in the schema (subject to cardinality constraints). In Figure 2(a), possible relational paths include [Actor] (a singleton path specifying an actor), [Movie, Stars-In, Actor] (specifying the actors in a movie), or even [Actor, Stars-In, Movie, Stars-In, Actor] (describing co-stars). The cardinality of a relational path is many if the cardinalities along the path indicate that it could reach more than one instance; otherwise, the cardinality is one. For example, *card*([Movie, Stars-In, Actor]) = many since a movie can reach many actors, whereas *card*([Actor]) = one since this path can only reach the base actor instance.

Relational variables consist of a relational path and an attribute, and they describe attributes of classes reached via a relational path (e.g., the popularity of actors starring in a movie). *Relational dependencies* consist of a pair of relational variables with a common first item, called the *perspective*. The dependency in Figure 2(a) states that the popularity of actors influences the success of movies they star in. A *canonical* dependency has a single item class in the relational path of the effect variable. A *relational model*, $\mathcal{M} = (\mathcal{S}, \mathcal{D})$, is a collection of relational dependencies $\mathcal{D}$, in canonical form, defined over schema $\mathcal{S}$. Relational models are parameterized by a set of conditional probability dis-



(a) Relational model (b) Relational skeleton (c) Ground graph

Figure 2: An example relational model involving actors and movies with a single relational dependency stating that actor popularity causes movie success. The variables in the ground graph are drawn from the instances in the skeleton, and the dependencies in the ground graph are drawn from the dependency in the model.

tributions, one for each attribute class $\mathcal{A}(I)$ for each $I \in \mathcal{E} \cup \mathcal{R}$, that factorizes a joint probability distribution for a given skeleton. This class of models can be expressed as DAPER models (Heckerman et al., 2007), and they are more general than plate models (Buntine, 1994; Gilks et al., 1994) and than PRMs with dependencies among only attributes (Getoor et al., 2007).

A relational model $\mathcal{M}$ paired with a relational skeleton σ produces a model instantiation $GG_{\mathcal{M}\sigma}$, called the *ground graph*. A ground graph is a directed graph with a node for each attribute of every entity and relationship instance in σ, and an edge between instances of relational variables for all dependencies in $\mathcal{M}$. A single relational model is a template for all possible ground graphs, one for every possible skeleton. Figure 2(c) shows an example ground graph. A ground graph has the same semantics as a Bayesian network with joint probability $P(GG_{\mathcal{M}\sigma}) = \prod_{I \in \mathcal{E} \cup \mathcal{R}} \prod_{X \in \mathcal{A}(I)} \prod_{i \in \sigma(I)} P(i.X \mid parents(i.X))$.

3.2 ABSTRACT GROUND GRAPHS

The RCD algorithm reasons about conditional independence using *abstract ground graphs*, introduced by Maier et al. (2013). Unlike the reasoning it supports in Bayesian networks, d-separation does not accurately infer conditional independence when applied directly to relational models. Abstract ground graphs enable sound and complete derivation of conditional independence facts using d-separation.

An abstract ground graph $AGG_{\mathcal{M}Bh}$ for relational model $\mathcal{M}$, perspective $B \in \mathcal{E} \cup \mathcal{R}$, and hop threshold $h \in \mathbb{N}^0$ is a directed graph that captures the dependencies among relational variables holding for any possible ground graph. $AGG_{\mathcal{M}Bh}$ has a node for each relational variable from perspective B with path length limited by h. $AGG_{\mathcal{M}Bh}$ contains edges between relational variables if the instantiations of those relational variables contain a dependent pair in some ground graph. Note that a *single* dependency in $\mathcal{M}$ may support *many* edges in $AGG_{\mathcal{M}Bh}$. Additionally, a *single* model $\mathcal{M}$ may produce *many* abstract ground graphs, one for each perspective.

Figure 1 shows abstract ground graphs for the model in Figure 2(a) from the ACTOR and MOVIE perspectives with $h = 4$. There is a single relational dependency in the example model, yet it supports two edges in each abstract ground graph. Also, one perspective exhibits a collider while the other contains a common cause. The abstract ground graph is the underlying representation used by RCD, and the conditional independence facts derived from it form the crux of the relational bivariate orientation rule.

4 EDGE ORIENTATION

Edge orientation rules, such as those used by the PC algorithm, use patterns of dependence and conditional independence to determine the direction of causality (Spirtes et al., 2000). In this section, we present the relational bivariate orientation rule and describe how the PC orientation rules can orient the edges of abstract ground graphs. We also prove that these orientation rules are individually sound and collectively complete for causally sufficient relational data.

4.1 BIVARIATE EDGE ORIENTATION

The example from Section 2 briefly describes the application of relational bivariate orientation (RBO). The abstract ground graph representation presents an opportunity to orient dependencies that cross relationships with a MANY cardinality. RBO requires no assumptions about functional form or conditional densities, unlike the recent work by Shimizu et al. (2006), Hoyer et al. (2008), and Peters et al. (2011) to orient bivariate dependencies. The only required assumption is the standard model acyclicity assumption, which restricts the space of dependencies to those without direct or indirect feedback cycles.

In the remainder of the paper, let I_W denote the item class on which attribute W is defined, and let $X - Y$ denote an undirected edge.

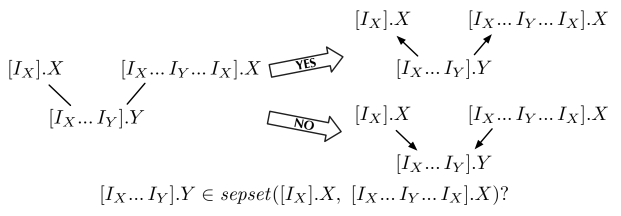

$[I_X].X$
$[I_X ... I_Y ... I_X].X$
$[I_X ... I_Y].Y$
YES
NO
$[I_X ... I_Y].Y \in sepset([I_X].X,\ [I_X ... I_Y ... I_X].X)$?

Figure 3: The relational bivariate orientation rule is conditional on whether $[I_X ... I_Y].Y$ is in the separating set of $[I_X].X$ and $[I_X ... I_Y ... I_X].X$.

Definition 1 (Relational Bivariate Orientation) Let $\mathcal{M}$ be a relational model and G a partially directed abstract ground graph for $\mathcal{M}$, perspective I_X, and hop threshold h. If $[I_X].X - [I_X ... I_Y].Y$ is in G, $card([I_Y ... I_X]) =$ MANY, and $[I_X].X \perp\!\!\!\perp [I_X ... I_Y ... I_X].X \mid \mathbf{Z}$, then (1) if $[I_X ... I_Y].Y \in \mathbf{Z}$, orient as $[I_X].X \leftarrow [I_X ... I_Y].Y$; (2) if $[I_X ... I_Y].Y \notin \mathbf{Z}$, orient as $[I_X].X \rightarrow [I_X ... I_Y].Y$.

RBO is illustrated in Figure 3. Given Definition 1, if $[I_X ... I_Y].Y$ is a collider for perspective I_X, then $[I_Y ... I_X].X$ is a common cause for perspective I_Y, assuming $card([I_Y ... I_X]) =$ MANY $= card([I_X ... I_Y])$. If $card([I_X ... I_Y]) =$ ONE and $card([I_Y ... I_X]) =$ MANY, then RBO applies only to the abstract ground graph with perspective I_X. For the example in Figure 1(a), [ACTOR, STARS-IN, MOVIE].*Success* is a collider for the ACTOR perspective.

RBO is akin to detecting relational autocorrelation (Jensen and Neville, 2002) and checking whether a distinct variable is a member of the set that eliminates the autocorrelation. It is also different than the collider detection rule (see Section 4.2) because it can explicitly orient dependencies as a common cause when the unshielded triple does not present itself as a collider. In Section 6.1, we quantify the extent to which RBO provides additional information beyond the standard PC edge orientation rules.

4.2 ORIENTING THE EDGES OF ABSTRACT GROUND GRAPHS

We adapt the rules for orienting edges in a Bayesian network, as used by PC (Spirtes et al., 2000) and characterized theoretically by Meek (1995), to orient relational dependencies at the level of abstract ground graphs. Figure 4 displays the four rules[1]—Collider Detection (CD), Known Non-Colliders (KNC), Cycle Avoidance (CA), and Meek Rule 3 (MR3)—as they would appear in an abstract ground graph.

[1]An additional rule is described by Meek (1995), but it only activates given prior knowledge.

A relational model has a corresponding set of abstract ground graphs, one for each perspective, but all are derived from the same relational dependencies. Recall from Section 3.2 that a single dependency supports many edges within and across the set of abstract ground graphs. Consequently, when a rule is activated for a *specific* abstract ground graph, the orientation of the underlying relational dependency must be propagated within and across *all* abstract ground graphs.

4.3 PROOF OF SOUNDNESS

An orientation rule is *sound* if any orientation not indicated by the rule introduces either (1) an unshielded collider in some abstract ground graph, (2) a directed cycle in some abstract ground graph, or (3) a cycle in the relational model (adapted from the definition of soundness given by Meek (1995)).

Theorem 1 *Let G be a partially oriented abstract ground graph from perspective B with correct adjacencies and correctly oriented unshielded colliders by either CD or RBO. Then, KNC, CA, MR3, and the purely common cause case of RBO, as well as the embedded orientation propagation, are sound.*

Proof. The proof for KNC, CA, and MR3 is nearly identical to the proof given by Meek (1995).

Orientation propagation: Let $[B ... I_X].X \rightarrow [B ... I_Y].Y$ be an oriented edge in G. By the definition of abstract ground graphs, this edge stems from a relational dependency $[I_Y ... I_X].X \rightarrow [I_Y].Y$. Let $[B ... I_X]'.X - [B ... I_Y].Y$ be an unoriented edge in G where $[B ... I_X]'$ is different than $[B ... I_X]$, but the edge is supported by the same underlying relational dependency. Assume for contradiction that the edge is oriented as $[B ... I_X]'.X \leftarrow [B ... I_Y].Y$. Then, there must exist a dependency $[I_X ... I_Y].Y \rightarrow [I_X].X$ in the model, which yields a cycle. The argument is the same for abstract ground graphs from different perspectives.

RBO common cause case: Given Definition 1, no alternate perspective would have oriented the triple as a collider, and $B = I_X$. Let $[I_X].X - [I_X ... I_Y].Y - [I_X ... I_Y ... I_X].X$ be an unoriented triple in G. Assume for contradiction that the triple is oriented as $[I_X].X \rightarrow [I_X ... I_Y].Y \leftarrow [I_X ... I_Y ... I_X].X$. This creates a new unshielded collider. Assume for contradiction that the triple is oriented as $[I_X].X \rightarrow [I_X ... I_Y].Y \rightarrow [I_X ... I_Y ... I_X].X$ or equivalently, the reverse direction. This implies a cycle in the model. ■

4.4 PROOF OF COMPLETENESS

A set of orientation rules is complete if it produces a maximally oriented graph. Any orientation of an un-

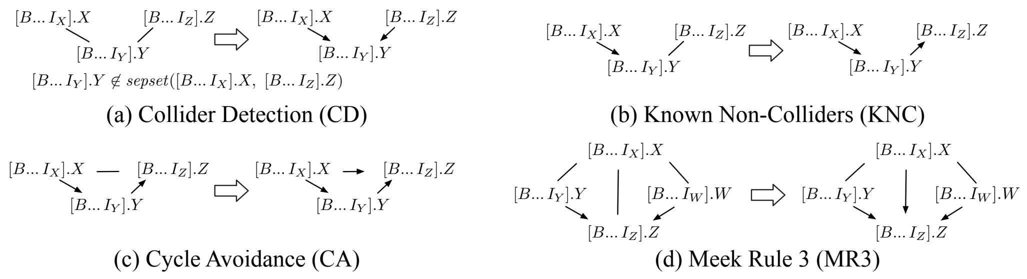

$[B...I_X].X$
$[B...I_Z].Z$
$[B...I_Y].Y$
$[B...I_Y].Y \notin sepset([B...I_X].X,\ [B...I_Z].Z)$
(a) Collider Detection (CD)
(b) Known Non-Colliders (KNC)
(c) Cycle Avoidance (CA)
$[B...I_W].W$
(d) Meek Rule 3 (MR3)

Figure 4: Schematics of the PC orientation rules on an abstract ground graph from perspective B.

oriented edge must be consistent with a member of the Markov equivalence class. Lemma 1 describes a useful property that enables the proof of completeness to reason directly about the remaining unoriented edges.

Lemma 1 *Let G be a partially oriented abstract ground graph, with correct adjacencies and oriented unshielded colliders. Let G_o be the result of exhaustively applying KNC, CA, MR3, and the purely common cause case of RBO all with orientation propagation. In G_o, if $P.X \to P'.Y - P''.Z$, then $P.X \to P''.Z$.*

Proof. Much of this proof follows from Meek (1995).

The following properties hold: (1) $X \neq Z$; otherwise, RBO would have oriented $P'.Y \leftarrow P''.Z$. (2) $P.X$ must be adjacent to $P''Z$; otherwise, KNC would have oriented $P'.Y \to P''.Z$. (3) $P.X \leftarrow P''.Z$ *does not* hold; otherwise, CA would have oriented $P'.Y \leftarrow P''.Z$. Therefore, we have a structure of the form $P.X \to P'.Y - P''.Z$ and $P.X - P''.Z$.

We show that $P.X \to P''.Z$ through exhaustive enumeration of the cases under which $P.X \to P'.Y$ was oriented. The cases for KNC, CD (and RBO collider cases), CA, and MR3 follow directly from Meek (1995).

(1) RBO oriented $P.X \to P'.Y$ from the I_Y perspective as a common cause. Then, $P' = [I_Y]$, $P = [I_Y ... I_X]$, and $P'' = [I_Y ... I_Z]$. Also, $[I_Y ... I_X ... I_Y].Y$ must be in G_o with $[I_Y ... I_X].X \to [I_Y ... I_X ... I_Y].Y$. By Definition 1, $card([I_Y ... I_X]) =$ ONE and $card([I_Y ... I_X]) =$ MANY.

The relational path $[I_Y ... I_Z]$ and its reverse have cardinality ONE; otherwise, RBO would have oriented $[I_Y].Y - [I_Y ... I_Z].Z$. We show that $[I_Y ... I_X].X - [I_Y ... I_Z].Z$ cannot remain unoriented.

Since this edge exists, by the construction of abstract ground graphs, (a) $[I_Y ... I_X]$ must be produced by combining $[I_Y ... I_Z]$ and $[I_Z ... I_X]$) (using the *extend* method (Maier et al., 2013)) and (b) $[I_Y ... I_Z]$ must be produced by combining $[I_Y ... I_X]$ and $[I_X ... I_Z]$). The paths $[I_X ... I_Z]$ and $[I_Z ... I_X]$ underlie the dependency between X and Z. Facts (a) and (b) impose constraints on the schema and abstract ground graphs. There are four cases for (a) depending on the relationship between $[I_Y ... I_Z]$ and $[I_Z ... I_X]$, with equivalent cases for (b).

(i) $[I_Y ... I_Z]$ and $[I_Z ... I_X]$ overlap exactly at I_Z. Then, the path from I_X to I_Z must have cardinality MANY. This implies that from the I_Z perspective, RBO would have oriented X to Z.
(ii) $[I_Y ... I_M ... I_Z]$ and $[I_Z ... I_M ... I_X]$ overlap up to I_M. This is equivalent to case (i), except I_M appears on the path from I_X to I_Z.
(iii) $[I_Z ... I_X]$ is a subpath of the reverse of $[I_Y ... I_Z]$. Then, the path from I_Z to I_Y must have cardinality MANY, which is a contradiction.
(iv) The reverse of $[I_Y ... I_Z]$ is a subpath of $[I_Z ... I_X]$. This is equivalent to case (i), except I_Y appears on the path from I_X to I_Z.

(2) Orientation propagation oriented $P.X \to P'.Y$. Then, there exists an edge for some perspective that was oriented by one of the orientation rules. From that perspective, the local structure matches the given pattern, and from the previous cases, $X \to Z$ was oriented. By definition, $P.X \to P''.Z$. ∎

Meek (1995) also provides the following results, used for proving completeness. A *chordal graph* is an undirected graph where every undirected cycle of length four or more has an edge between two nonconsecutive vertices on the cycle. Let G be an undirected graph, α a total order on the vertices of G, and G_α the induced directed graph ($A \to B$ is in G_α if and only if $A < B$ with respect to α). A total order α is *consistent* with respect to G if and only if G_α has no unshielded colliders. It can be shown that only chordal graphs have consistent orderings. Finally, if G is an undirected chordal graph, then for all pairs of adjacent vertices A and B in G, there exist consistent total orderings α and γ such that $A \to B$ in G_α and $A \leftarrow B$ in G_γ.

Theorem 2 *Given a partially oriented abstract ground graph, with correct adjacencies and oriented* *unshielded colliders, exhaustively applying KNC, CA, MR3, and RBO all with orientation propagation results in a maximally oriented graph G.*

Proof. Much of this proof follows from Meek (1995). Let E_u and E_o be the set of unoriented edges and oriented edges of G, respectively.

Claim 1: No orientation of edges in E_u creates a cycle or unshielded collider in G that includes edges from E_o.
Proof. Assume there exists an orientation of edges in E_u that creates a cycle using edges from E_o. Without loss of generality, assume that the cycle is of length three. (1) If $A \to B \to C$ are in E_o and $A - C$ in E_u, then CA would have oriented $A \to C$. (2) If $A \to B \leftarrow C$ or $A \leftarrow B \to C$ are in E_o and $A - C$ is in E_u, then no orientation $A - C$ would create a cycle. (3) If $A \to B$ is in E_o and $B - C - A$ in E_u, then by Lemma 1 we have $A \to C$ and no orientation of $B - C$ would create a cycle. A similar argument holds for unshielded colliders. □
Claim 2: Let G_u be the subgraph of G containing only unoriented edges. G_u is the union of disjoint chordal graphs.
Proof. Assume that G_u is not the union of disjoint chordal graphs. Then, there exists at least one disjoint component of G_u that is not a chordal graph. Recall that no total ordering of G_u is consistent. Let $A \to B \leftarrow C$ be an unshielded collider induced by some ordering on G_u. There are two cases: (1) A and C are adjacent in G. The edge must be oriented; otherwise, it would appear in G_u. Both orientations of $A - C$ imply an orientation of A and B, or C and B, by Lemma 1. (2) A and C are not adjacent in G. Then, $A - B - C$ is an unshielded triple in G. Either CD or RBO would have oriented the triple as a collider, or the triple is inconsistent with the total ordering on G_u. □

Since G is chordal, it follows that no orientation of the unoriented edges in G creates a new unshielded collider or cycle. ■

5 The RCD Algorithm

The relational causal discovery (RCD) algorithm is a sound and complete constraint-based algorithm for learning causal models from relational data.[2] RCD employs a similar strategy to the PC algorithm, operating in two distinct phases (Spirtes et al., 2000). RCD is similar to the Relational PC (RPC) algorithm, which also learns causal relational models (Maier et al., 2010). The differences between RPC and RCD are threefold: (1) The underlying representation for RCD is a set of abstract ground graphs; (2) RCD employs a new causal constraint—the relational bivariate orientation rule; and (3) RCD is sound and complete. RPC also reasons about the uncertainty of relationship existence, but RCD assumes a prior relational skeleton.

ALGORITHM 1: RCD($schema, depth, hopThreshold, P$)

```
1  $PDs \leftarrow$ getPotentialDeps($schema, hopThreshold$)
2  $N \leftarrow$ initializeNeighbors($schema, hopThreshold$)
3  $S \leftarrow \{\}$
   // Phase I
4  for $d \leftarrow 0$ to $depth$ do
5      for $X \to Y \in PDs$ do
6          foreach $condSet \in$ powerset($N[Y] \setminus \{X\}$) do
7              if $|condSet| = d$ then
8                  if $X \perp\!\!\!\perp Y \mid condSet$ in $P$ then
9                      $PDs \leftarrow PDs \setminus \{X \to Y, Y \to X\}$
10                     $S[X, Y] \leftarrow condSet$
11                     break
   // Phase II
12 $AGGs \leftarrow$ buildAbstractGroundGraph($PDs$)
13 $AGGs, S \leftarrow$ ColliderDetection($AGGs, S$)
14 $AGGs, S \leftarrow$ BivariateOrientation($AGGs, S$)
15 while $changed$ do
16     $AGGs \leftarrow$ KnownNonColliders($AGGs, S$)
17     $AGGs \leftarrow$ CycleAvoidance($AGGs, S$)
18     $AGGs \leftarrow$ MeekRule3($AGGs, S$)
19 return getCanonicalDependencies($AGGs$)
```

The remainder of this section describes the algorithmic details of RCD and proves its correctness.

Algorithm 1 provides pseudocode for RCD. Initially, RCD enumerates the set of potential dependencies, in canonical form, with relational paths limited by the hop threshold (line 1). Phase I continues similarly to PC, removing potential dependencies via conditional independence tests with conditioning sets of increasing size drawn from the power set of neighbors of the effect variable (lines 4–11). Every identified separating set is recorded, and the corresponding potential dependency and its reverse are removed (lines 9–10).

The second phase of RCD determines the orientation of dependencies consistent with the conditional independencies discovered in Phase I. First, Phase II constructs a set of undirected abstract ground graphs, one for each perspective, given the remaining dependencies. RCD then iteratively checks all edge orientation rules, as described in Section 4. Phase II of RCD is also different from PC and RPC because it searches for additional separating sets while finding colliders and common causes with CD and RBO. Frequently, unshielded triples $X - Y - Z$ may have no separating set recorded for X and Z. For these pairs, RCD attempts to discover a new separating set, as in Phase I. These triples occur for one of three reasons: (1) Since X and Z are relational variables, the separating set may have been discovered from an alternative perspective; (2) The total number of hops in the relational paths for X, Y, and Z may exceed the hop threshold—each dependency is subject to the hop threshold, but a pair of

[2]Code available at kdl.cs.umass.edu/rcd.

dependencies is limited by twice the hop threshold; or (3) The attributes of relational variables X and Z are the same, which is necessarily excluded as a potential dependency by the assumption of an acyclic model.

Given the algorithm description and the soundness and completeness of the edge orientation rules, we prove that RCD is sound and complete. The proof assumes causal sufficiency and a prior relational skeleton (i.e., no causes of the relational structure).

Theorem 3 *Given a schema and probability distribution P, RCD learns a correct maximally oriented model $\mathcal{M}$ assuming perfect conditional independence tests, sufficient hop threshold h, and sufficient depth.*

Proof sketch. Given sufficient h, the set of potential dependencies PDs includes all true dependencies in $\mathcal{M}$, and the set of neighbors N includes the true causes for every effect relational variable. Assuming perfect conditional independence tests, PDs includes exactly the undirected true dependencies after Phase I, and $S[X, Y]$ records a correct separating set for the relational variable pair $\langle X, Y \rangle$. However, there may exist non-adjacent pairs of variables that have no recorded separating set (for the three reasons mentioned above). Given the remaining dependencies in PDs, we construct the correct set of edges in $AGGs$ using the methods from Maier et al. (2013). Next, all unshielded colliders are oriented by either CD or RBO, with correctness following from Spirtes et al. (2000) and relational d-separation (Maier et al., 2013). Whenever a pair $\langle X, Y \rangle$ is missing a separating set in S, it is either found as in Phase I or from a different perspective. RCD then produces a maximally oriented model by the soundness (Theorem 1) and completeness (Theorem 2) results of the remaining orientation rules. ■

6 EXPERIMENTS

6.1 SYNTHETIC EXPERIMENTS

The proofs of soundness and completeness offer a qualitative measure of RCD's effectiveness—no other method can learn a more accurate causal model from observational data. To complement the theoretical results, we provide a *quantitative* measure of RCD's performance and compare against the performance of alternative constraint-based algorithms.

We evaluate RCD against two alternative algorithms. The first algorithm is RPC (Maier et al., 2010). This provides a comparison against current state-of-the-art relational learning. The second algorithm is the PC algorithm executed on relational data that has been propositionalized from a specific perspective—termed Propositionalized PC (PPC). Propositionalization reduces relational data to a single, propositional table (Kramer et al., 2001). We take the best and worst perspectives for each trial by computing the average F-score of its skeleton and compelled models.

We generated 1,000 random causal models over randomly generated schemas for each of the following combinations: entities (1–4); relationships (one less than the number of entities) with cardinalities selected uniformly at random; attributes per item drawn from $Pois(\lambda = 1) + 1$; and relational dependencies (1–15) limited by a hop threshold of 4 and at most 3 parents per variable. This procedure yielded a total of 60,000 synthetic models. Note that this generates simple Bayesian networks when there is a single entity class. We ran RCD, RPC, and PPC for each perspective, using a relational d-separation oracle with hop threshold 8 for the abstract ground graphs.

We compare the learned causal models with the true causal model. For each trial, we record the precision (the proportion of learned edges in the true model) and recall (the proportion of true edges in the learned model) for both the undirected skeleton after Phase I and the partially orientated model after Phase II. Figure 5 displays the average across 1,000 trials for each algorithm and measure. We omit error bars as the maximum standard error was less than 0.015.

All algorithms learn identical models for the single-entity case because they reduce to PC when analyzing propositional data. For truly relational data, algorithms that reason over relational representations are necessary for accurate learning. RCD and RPC recover the exact skeleton, whereas the best and worst PPC cases learn flawed skeletons (and also flawed oriented models), with high false positive and high false negative rates. This is evidence that propositionalizing relational data may lead to inaccurately learned causal models.

For oriented models, the RCD algorithm vastly exceeds the performance of all other algorithms. As the soundness result suggests, RCD achieves a compelled precision of 1.0, whereas RPC introduces orientation errors due to reasoning over the class dependency graph and missing additional separating sets. For recall, which is closely tied to the completeness result, RCD ranges from roughly 0.56 (for 1 dependency and 2 entities) to 0.94 (for 15 dependencies and 4 entities). While RPC and PPC cannot orient models with a single dependency, the relational bivariate orientation rule enables RCD to orient models using little information. RCD also discovers more of the underlying causal structure as the complexity of the domain increases, with respect to both relational structure (more entity and relationship classes) and model density.

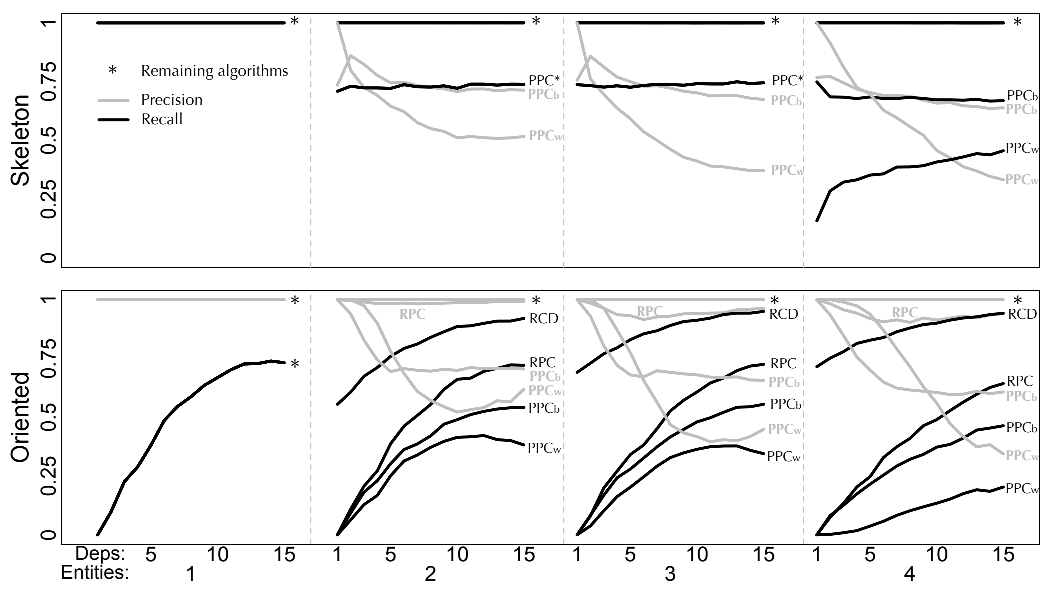


Figure 5: Skeleton and oriented precision and recall for the RCD and RPC algorithms, as well as the best and worst perspective for PPC for a baseline. Results are averaged over 1,000 models for each setting.

To quantify the unique contribution that RBO provides, we applied RBO as the *final* orientation rule in Phase II and recorded the frequency with which each edge orientation rule is activated (see Figure 6). As expected, RBO never activates for the single-entity case because all paths have cardinality ONE. For truly relational domains, RBO orients between 11% and 100% of the oriented edges. However, this does not fully capture the broad applicability of RBO. Therefore, we also recorded the frequency of each edge orientation rule when RBO is applied *first* in Phase II of RCD. In this case, for at least two entity classes, RBO orients between 58% and 100% of the oriented edges.

Finally, we recorded the number of conditional independence tests used by the RCD and RPC algorithms. RCD learns a more accurate model than RPC, but at the cost of running additional tests of independence during Phase II. Fortunately, these extra tests do not alter the asymptotic complexity of the algorithm, requiring on average 31% more tests.

6.2 DEMONSTRATION ON REAL DATA

We applied RCD to the MovieLens+ database, a combination of the UMN MovieLens database (www.grouplens.org); box office, director, and actor information collected from IMDb (www.imdb.com); and average critic ratings from Rotten Tomatoes (www.rottentomatoes.com). Of the 1,733 movies with this additional information, we sampled 10% of the user ratings yielding roughly 75,000 ratings. For testing conditional independence, RCD checks the significance of coefficients in linear regression and uses the average aggregation function for relational variables. The RCD-generated model is displayed in Figure 7.

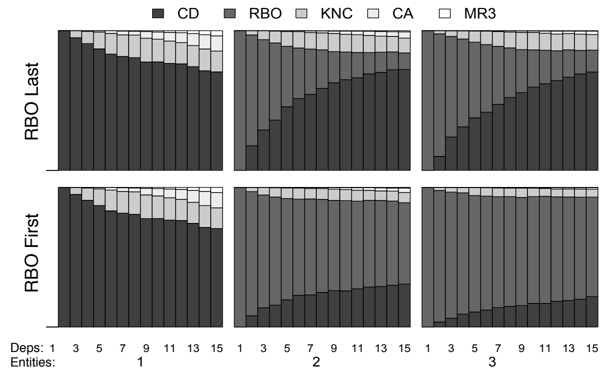


Figure 6: Frequency of edge orientation rules in RCD, with RBO last (above) and first (below).

We ran RCD with a hop threshold of 4, maximum depth of 3, and an effect size threshold of 0.01. Because constraint-based methods are known to be order-dependent (Colombo and Maathuis, 2012), we ran RCD 100 times and used a two-thirds majority vote to determine edge presence and orientation. RCD discovered 27 dependencies. One interesting dependency is that the average number of films that actors have starred in affects the number of films the director has directed—perhaps well-established actors tend to work with experienced directors. Also, note that genre is a composition of binary genre attributes.

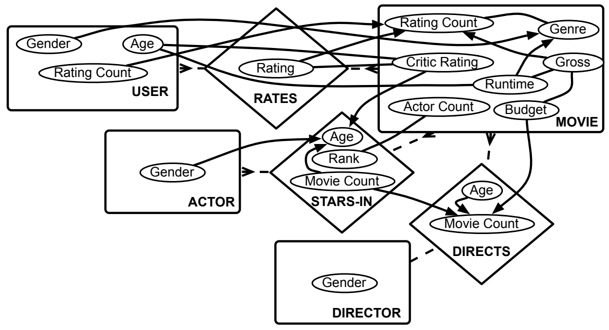


Figure 7: RCD-learned model of MovieLens+.

7 RELATED WORK

The ideas presented in this paper are related to three primary areas of research. First, the RCD algorithm is a constraint-based method for learning causal structure from observed conditional independencies. The vast majority of other causal discovery algorithms have focused on Bayesian networks and propositional data. The IC (Pearl, 2000) and PC (Spirtes et al., 2000) algorithms provided a foundation for all future constraint-based methods, and Meek (1995) proved these equivalent methods to be sound and complete for causally sufficient data. Additional constraint-based methods include the Grow-Shrink (Margaritis and Thrun, 1999) and TC (Pellet and Elisseeff, 2008) algorithms.

Second, RCD emphasizes learning causal *relational* models, a more expressive class of models for real-world systems. Our experimental results also indicate that propositional approaches may be inadequate to handle the additional complexity of relational data. Algorithms for learning the structure of directed relational models have been limited to methods based on search-and-score that do not identify Markov equivalence classes (Friedman et al., 1999). The RPC algorithm was the first to employ constraint-based methods to learn causal models from relational data (Maier et al., 2010), but RPC is not complete and may introduce errors due to its underlying representation. Both RPC and PRMs include capabilities to reason about relationship existence (Getoor et al., 2002); however, we currently focus on attributional dependencies and leave causes of existence as future work.

Finally, orienting bivariate dependencies, the most effective constraint used by RCD, is similar to the efforts of Shimizu et al. (2006), Hoyer et al. (2008), and Peters et al. (2011) in the propositional setting. Contrary to RBO, these techniques leverage strong assumptions on functional form and asymmetries in conditional densities to determine the direction of causality. Nonetheless, these methods could orient some of the edges that remain unoriented by RCD, given these additional distributional assumptions.

8 CONCLUSIONS

Relational d-separation and the abstract ground graph representation provide a new opportunity to develop theoretically correct algorithms for learning causal structure from relational data. We presented the relational causal discovery (RCD) algorithm and proved it sound and complete for discovering causal models from causally sufficient relational data. We introduced relational bivariate orientation (RBO), which can detect the orientation of bivariate dependencies. This leads to recall of oriented relational models over a previous state-of-the-art algorithm that is 18% to 72% greater on average. We also demonstrated RCD's effectiveness on synthetic causal relational models and demonstrated its applicability to real-world data.

There are several clear avenues for future research. RCD could be extended to reason about entity and relationship existence, and the assumptions of causal sufficiency and acyclic models could be relaxed to support reasoning about latent common causes and temporal dynamics. There are also new operators that exploit relational structure, such as relational blocking (Rattigan et al., 2011), which could be integrated with simple tests of conditional independence. Finally, RCD could be enhanced with Bayesian information, similar to the recent work by Claassen and Heskes (2012) for improving the reliability of algorithms that learn the structure of Bayesian networks.

Acknowledgments

The authors wish to thank Cindy Loiselle for her editing expertise. This effort is supported by the Intelligence Advanced Research Project Agency (IARPA) via Department of Interior National Business Center Contract number D11PC20152, Air Force Research Lab under agreement number FA8750-09-2-0187, the National Science Foundation under grant number 0964094, and Science Applications International Corporation (SAIC) and DARPA under contract number P010089628. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of IARPA, DoI/NBC, AFRL, NSF, SAIC, DARPA or the U.S. Government. The Greek State Scholarships Foundation partially supported Katerina Marazopoulou.

References

W. L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, 2:159–225, 1994.

T. Claassen and T. Heskes. A Bayesian approach to constraint based causal inference. In *Proceedings of the Twenty-Eighth Annual Conference on Uncertainty in Artificial Intelligence*, pages 207–216, 2012.

D. Colombo and M. H. Maathuis. A Modification of the PC Algorithm Yielding Order-Independent Skeletons. *arXiv preprint arXiv:1211.3295*, 2012.

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *International Joint Conference on Artificial Intelligence*, volume 16, pages 1300–1309, 1999.

L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of link structure. *Journal of Machine Learning Research*, 3:679–707, 2002.

L. Getoor, N. Friedman, D. Koller, A. Pfeffer, and B. Taskar. Probabilistic relational models. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*, pages 129–174. MIT Press, Cambridge, MA, 2007.

W. R. Gilks, A. Thomas, and D. J. Spiegelhalter. A language and program for complex Bayesian modeling. *The Statistician*, 43:169–177, 1994.

D. Heckerman, C. Meek, and D. Koller. Probabilistic entity-relationship models, PRMs, and plate models. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*, pages 201–238. MIT Press, Cambridge, MA, 2007.

P. O. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems 22*, pages 689–696, 2008.

D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 259–266, 2002.

S. Kramer, N. Lavrač, and P. Flach. Propositionalization approaches to relational data mining. In S. Džeroski and N. Lavrač, editors, *Relational Data Mining*, pages 262–286. Springer-Verlag, New York, NY, 2001.

M. Maier, B. Taylor, H. Oktay, and D. Jensen. Learning causal models of relational domains. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 531–538, 2010.

M. Maier, K. Marazopoulou, and D. Jensen. Reasoning about Independence in Probabilistic Models of Relational Data. *arXiv preprint arXiv:1302.4381*, 2013.

D. Margaritis and S. Thrun. Bayesian network induction via local neighborhoods. In *Advances in Neural Information Processing Systems 12*, pages 505–511, 1999.

C. Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 403–410, 1995.

J. Pearl. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, New York, NY, 2000.

J.-P. Pellet and A. Elisseeff. Using Markov blankets for causal structure learning. *Journal of Machine Learning Research*, 9:1295–1342, 2008.

J. Peters, D. Janzing, and B. Schölkopf. Causal inference on discrete data using additive noise models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2436–2450, 2011.

M. J. Rattigan, M. Maier, and D. Jensen. Relational blocking for causal discovery. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pages 145–151, 2011.

S. Shimizu, P. O. Hoyer, A. Hyvarinen, and A. Kerminen. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search.* MIT Press, Cambridge, MA, 2nd edition, 2000.

Evaluating Anytime Algorithms for Learning Optimal Bayesian Networks

Brandon Malone
Department of Computer Science
Helsinki Institute for Information Technology
Fin-00014 University of Helsinki, Finland
brandon.malone@cs.helsinki.fi

Changhe Yuan
Department of Computer Science
Queens College/City University of New York
Queens, NY 11367 USA
changhe.yuan@qc.cuny.edu

Abstract

Exact algorithms for learning Bayesian networks guarantee to find provably optimal networks. However, they may fail in difficult learning tasks due to limited time or memory. In this research we adapt several anytime heuristic search-based algorithms to learn Bayesian networks. These algorithms find high-quality solutions quickly, and continually improve the incumbent solution or prove its optimality before resources are exhausted. Empirical results show that the anytime window A* algorithm usually finds higher-quality, often optimal, networks more quickly than other approaches. The results also show that, surprisingly, while generating networks with few parents per variable are structurally simpler, they are harder to learn than complex generating networks with more parents per variable.

1 INTRODUCTION

Score-based learning of Bayesian networks is a popular strategy which assigns a score to a network structure based on given data, and the goal is to find the highest-scoring structure. The problem is NP-complete (Chickering 1996), so much early research focused on local search strategies, such as greedy hill climbing in the space of Bayesian network structures (Heckerman 1996), hill climbing in the space of equivalence classes of networks (Chickering 2002) or hill climbing in the space of variable orderings (Teyssier and Koller 2005). Other more sophisticated local search techniques have also been investigated (Moore and Wong 2003). Unfortunately, these algorithms offer no bounds on the quality of learned networks. On the other hand, they do have good *anytime* behavior. That is, they quickly find a solution and improve its quality throughout the search. The search can be stopped at "anytime" and return the best solution found so far.

Several dynamic programming (DP) algorithms (Koivisto and Sood 2004; Ott, Imoto, and Miyano 2004; Singh and Moore 2005; Silander and Myllymaki 2006; Malone, Yuan, and Hansen 2011) have been developed which guarantee to find the highest scoring network for a dataset. However, these algorithms do not exhibit anytime behavior; they do not produce any solution until giving the optimal network at the end of the search.

Recently, though, several algorithms have been developed which include both optimality guarantees and anytime behavior. de Campos and Ji (2011) proposed a branch and bound algorithm (BB). It begins with a (cyclic) structure in which all variables have their optimal parents. Then, cycles are broken in a best-first manner until the optimal structure is found. These cyclic structures give a lower bound on the optimal network which improves throughout the search. To add anytime behavior, a local search algorithm is used to learn a suboptimal network at the beginning of the search. The score of that network serves as an upper bound. Furthermore, the search sometimes deviates from a pure best-first strategy to find acyclic structures and improve the upper bound. At anytime, the search can be stopped, and the ratio between the upper and lower bounds give a quality guarantee of the current best acyclic network. When the two bounds agree, the current best structure is optimal.

Mathematical programming (MP) algorithms (Jaakkola et al. 2010; Cussens 2011) have also been developed which have both anytime behavior and optimality guarantees. These algorithms search in a space which includes an embedded polytope whose surface corresponds to Bayesian networks. The polytope has exponentially many facets, so it is not represented explicitly. Rather, a series of MPs are solved to define the polytope and find the optimal point on its surface, which corresponds to the optimal Bayesian network. The points on the surface correspond to integer coordinates, so the MPs are actually integer linear programs (ILPs) which are solved by relaxing the problem to a normal linear program (LP). After solving each LP, the solution is checked for integrality. If it is integral, then the solution corresponds to the optimal Bayesian network. If not, the value of the solution gives a lower bound on the optimal score. Also, the solution can be used to decode a valid acyclic network and find an upper bound on the score. As with BB, at any time, the search can be stopped, and the ratio between the bounds gives a quality guarantee.

Yuan *et al.* (2011) described a shortest path formulation for the structure learning problem. Since then, several heuristic search algorithms, including A* (Yuan, Malone, and Wu 2011) and BFBnB (Malone et al. 2011), have been applied to this problem. This paper explores the empirical behavior of a variety of *anytime heuristic search algorithms*, including anytime weighted A* (AWeiA*) (Hansen and Zhou 2007), anytime repairing A* (ARA*) (Likhachev, Gordon, and Thrun 2003) and anytime window A* (AWinA*) (Aine, Chakrabarti, and Kumar 2007), within this shortest path formulation. Like BB and MP, these algorithms all incorporate optimality guarantees and anytime behavior. We empirically compare these algorithms and MP on a variety of synthetic datasets. We use synthetic datasets because these allow us to better control experimental conditions which affect the learning, including the number of variables, number of records and complexity of the generating process of the datasets.

Experimentally, we show that AWinA* outperforms the other anytime heuristic search algorithms. It also often finds higher-quality networks more quickly than MP, but is slower to prove optimality for simpler synthetic networks. More thorough investigation into the search space and runtime characteristics of the algorithms provide additional insight to the learning problem. In particular, our results show that complex generating networks may seem structurally challenging to learn, but they actually lie within or close to the promising solution space that is first explored by heuristic search and are thus easier to find. In contrast, simple generating networks typically receive bad estimated scores. Because many other search nodes have better score estimates, heuristic search cannot easily prove optimality for these datasets.

2 LEARNING BAYESIAN NETWORKS

A Bayesian network consists of a directed acyclic graph (DAG) in which vertices correspond to a set of random variables $\boldsymbol{V} = \{X_1, ..., X_n\}$ and a set of conditional probability distributions. The arcs in the DAG encode conditional independence relations among the variables. We use PA_i to represent the parent set of X_i. The dependence between each variable X_i and its parents is quantified using a conditional probability distribution, $P(X_i|PA_i)$. The product of the conditional probability distributions give the joint distribution over all of the variables.

We consider the score-based Bayesian network structure learning problem (BNSL) in this paper. Given a dataset $\boldsymbol{D} = \{D_1, ..., D_N\}$, where D_i is a complete instantiation of all of the variables $\mathbf{V}$, and a scoring function s, the goal is to find a Bayesian network structure S^* such that $S^* = \arg\min_S s(S, \mathbf{D})$. We omit $\mathbf{D}$ for brevity in the remainder of the paper.

The scoring function is often a penalized log-likelihood or Bayesian criterion which trades off the goodness of fit of S to $\mathbf{D}$ against the complexity of S. We allow for any *decomposable* score, i.e., the score for S is the sum of the scores of each variable, $s(S) = \sum_{X \in \mathbf{V}} s(X, PA_X)$. Most commonly used scoring functions, including MDL (Lam and Bacchus 1994), fNML (Silander et al. 2008) and BDe (Buntine 1991; Heckerman 1996), are decomposable.

In our work, we adopt a shortest path perspective to the problem, so we assume the optimal structure minimizes s. Some scoring functions, such as BDe, assign high values to better networks. We can multiply all scores by -1 to convert the maximization into a minimization.

3 THE SHORTEST PATH PERSPECTIVE

Yuan *et al.* (2011) formulated BNSL as a shortest-path finding problem. Figure 1 shows an implicit search graph for four variables in which the shortest path search is performed. Each node in the graph corresponds to an optimal subnetwork over a unique subset of variables in the dataset. The *start* search node, at the top of the graph, corresponds to the empty variable set, while the bottom-most node with all variables is the *goal* node. Each edge in the search graph represents adding a new variable X as a leaf to the optimal subnetwork over the existing variables, $\mathbf{U}$. The new variable selects its optimal parents (according to the scoring function s) from $\mathbf{U}$. The cost of the edge is equal to the score of the optimal parent set, which we denote $BestScore(X, \mathbf{U})$, i.e.,

$$\begin{aligned} cost(\mathbf{U} \rightarrow \mathbf{U} \cup \{X\}) &= BestScore(X, \mathbf{U}) \\ &= \min_{PA_X \subseteq \mathbf{U}} s(X, PA_X). \end{aligned}$$

Based on this specification, a path from $start$ to $goal$ induces an ordering on the variables, based on the order in which they are added. Thus, we also call this graph the *order graph*. Each variable selects its optimal parents from variables which precede it in the ordering. Consequently, combining the parent set selections made on a path from $start$ to $goal$ gives the optimal network for that ordering, and the cost of that path corresponds to the score for that network. Therefore, the shortest path from $start$ to $goal$ corresponds to a globally optimal Bayesian network.

The computation of $BestScore(\cdot)$ is required for each edge visited during the search. Naively, this computation requires considering an exponential number of parent sets; however, several authors (Teyssier and Koller 2005; de Campos and Ji 2011) have noted that many parent sets are not optimal for any ordering of variables. Therefore, many local scores can be pruned before the search. Yuan

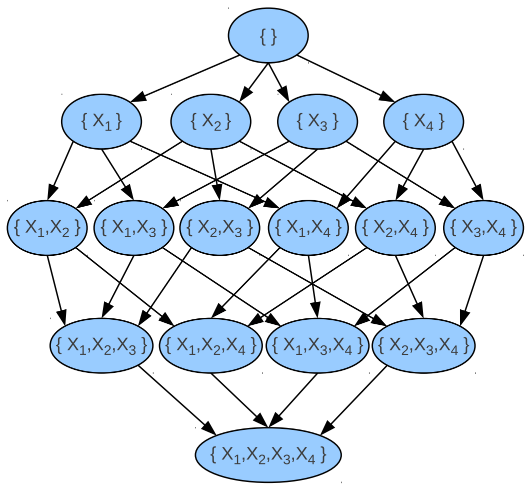


Figure 1: An order graph of four variables

and Malone (2012) developed a sparse data structure which takes advantage of this pruning to store only the *possibly optimal parent sets* (POPS) and to compute $BestScore(\cdot)$ with a linear number of bitwise operations.

This shortest path problem has been solved using several heuristic search algorithms, including A* (Yuan, Malone, and Wu 2011) 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 $\mathbf{U}$ in the order graph to $goal$. An f-cost is calculated for $\mathbf{U}$ by summing the cost from $start$ to $\mathbf{U}$ (called $g(\mathbf{U})$) and the lower bound from $\mathbf{U}$ to $goal$ (called $h(\mathbf{U})$). So $f(\mathbf{U}) = g(\mathbf{U}) + h(\mathbf{U})$. The f-cost provides an optimistic estimation on how good a path can be if it has to go through $\mathbf{U}$. The search maintains a list of nodes to be expanded sorted by f-costs called $open$. It also keeps a list of nodes which have already been expanded called $closed$. Initially, $open$ contains just $start$, and $closed$ is empty. Nodes are then expanded 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 $open$. In general, if a better path to a node in $closed$ is found, then the node must be added to $open$ again and re-expanded. Upon expanding $goal$, the shortest path from $start$ to $goal$ has been found.

In BFBnB, nodes are instead expanded one layer at a time, where a layer consists of all nodes corresponding to subnetworks of the same size. Before beginning the search, a local search strategy, such as greedy hill climbing, is used to find a "good" network and its score. During the BFBnB search, any node with an f-cost greater than the score found during the local search can safely be pruned.

Yuan *et al.* (Yuan, Malone, and Wu 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 $\mathbf{U}$ to $goal$. Furthermore, the heuristics have been shown to be *consistent*, which is a property similar to non-negativity required by Dijkstra's algorithm. Primarily, in standard 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.

4 ANYTIME LEARNING ALGORITHMS

The shortest path perspective makes it straightforward to apply anytime heuristic search algorithms to solve the Bayesian network learning problem. The basic A* algorithm does not have anytime behavior. It expands nodes in best-first order until expanding $goal$ at which point it has the optimal solution. The heuristic search community has developed a variety of algorithms which allow A* to find solutions more quickly. We begin by discussing weighted A* (WA*), which does not add anytime behavior to A* but can greatly improve solving time while offering provable quality guarantees. We then discuss two techniques which add anytime behavior to WA*. We also describe a third strategy, not directly related to WA*, which adds anytime behavior and quality guarantees to A*.

4.1 WEIGHTED A*

Weighted A* (WA*) (Pohl 1970) is a variant of A* which adds a weight ϵ (≥ 1.0) to the heuristic function in the f-cost calculations. That is, $f(\mathbf{U}) = g(\mathbf{U}) + \epsilon \times h(\mathbf{U})$. Otherwise, WA* behaves exactly as unweighted A*. The inflated h-cost could now overestimate the cost of a path from $\mathbf{U}$ to the goal, so the calculation is no longer admissible or consistent. Despite the loss of admissibility, though, WA* still has a quality guarantee: if h was originally consistent, the search algorithm can disregard any better paths it finds to nodes in $closed$, and the cost of the path found from $start$ to $goal$ is guaranteed to be no more than a factor of ϵ greater than the optimal solution.

Intuitively, much of the f-cost of nodes close to $start$ comes from h, but the f-cost of deeper nodes is dominated by g. Because WA* weights the h costs, but not the g costs, this has the effect of making the search favor deeper nodes because they have smaller h values. Thus, the overall effect is that WA* expands deeper nodes more greedily than A* and often expands $goal$ much more quickly than the unweighted A*.

4.2 ANYTIME WEIGHTED A*

Like WA*, anytime weighted A* (AWeiA*) (Hansen and Zhou 2007) adds a weight to the heuristic calculations so it also favors expanding deeper nodes. Rather than stopping as soon as $goal$ is expanded, though, AWeiA* continues the search. During the search, a stream of better paths to $goal$ are discovered, and the *incumbent* solution, which gives the current shortest path, is updated. If the search is run until completion, it terminates with the optimal solution. To guarantee the optimality of the final solution, though, AWeiA* must re-expand nodes when it finds a better path to them. Any node which has a worse unweighted f-cost than the incumbent can be pruned, though. At any time, the search can be stopped, and the incumbent solution returned. AWeiA* also offers the same guarantee of WA*: the globally optimal solution is guaranteed to be within a factor of ϵ of the incumbent. However, an even tighter error bound is available by calculating the ratio of the smallest unweighted f-cost of any open node and the incumbent.

4.3 ANYTIME REPAIRING A*

Anytime repairing A* (ARA*) (Likhachev, Gordon, and Thrun 2003) also starts as normal WA*. Upon finding a solution, ARA* decreases ϵ and searches again. At each iteration, the solution improves (or stays the same), so this algorithm also produces a stream of improved solutions. Additionally, because ϵ is decreased at each iteration, the quality guarantee tightens, as well. ARA* can also check the ratio between the smallest unweighted f-cost and the incumbent to look for an even better bound. The algorithm terminates with the optimal solution after completing an iteration in which $\epsilon = 1$.

Like AWeiA*, ARA* can also find a better path to a node during the search. Rather than immediately adding the node back to $open$, though, ARA* keeps these nodes in a separate list, $repair$. At the beginning of each iteration, rather than beginning the search at $start$, ARA* instead adds all of $repair$ to $open$. In this manner, ARA* reuses g-cost information from one iteration to the next. ARA* can also prune nodes with a worse f-cost than the incumbent.

4.4 ANYTIME WINDOW A*

Unlike AWeiA* and ARA*, anytime window A* (AWinA*) (Aine, Chakrabarti, and Kumar 2007) is not based on WA*. Rather, it uses a type of sliding window to encourage deeper exploration of the order graph. It consists of a series of iterations in which a parameter w, which increases from one iteration to the next, controls the size of the window. The algorithm keeps track of the depth of all nodes expanded during an iteration of the algorithm. After expanding a node in layer l, all nodes in layer $l - w$ are *frozen* for that iteration. Frozen nodes are stored, but are not expanded. When h is consistent (like the heuristic functions used here in BNSL), AWinA* will only expand a node at most once during each iteration. Therefore, node re-expansions are not explicitly considered in this algorithm. As with the other anytime algorithms, AWinA* can prune any node with a worse f-cost than the incumbent. Similar to ARA*, rather than starting each iteration at $start$, AWinA* begins by adding all frozen nodes from the previous iteration to $open$, so it also reuses information across iterations.

AWinA* uses the sliding window to encourage more greedy behavior in the search, but there is no quality guarantee for window size similar to that of the weighted algorithms. This algorithm can calculate the ratio between the smallest unweighted f-cost of a frozen node and the incumbent to find a quality guarantee, though.

5 EMPIRICAL EVALUATIONS

We empirically evaluated the anytime weighted A* (AWeiA*), anytime window A* (AWinA*), and anytime repairing A* (ARA*) against the integer linear programming algorithm (GOBNILP, v1.1) (Cussens 2011) on Bayesian network learning tasks. The A* implementations are available online (http://url.cs.qc.cuny.edu/software/URLearning.html). For AWeiA*, we used a weight of 1.25. For ARA*, we also set the initial weight to be 1.25 and decreased it by 0.05 at each iteration. The initial window size of AWinA* was 0 and increased by 1 after each iteration. We used static pattern databases as the heuristic function. We empirically determined these parameters give good performance on a variety of datasets. We used the default parameter setting for the GOBNILP algorithm (ILP for short). We did not compare to local search strategies, such as greedy hill climbing or optimal reinsertion, because a previous study (Malone and Yuan 2012) has shown that WA* outperforms those algorithms. That study also showed that WA* outperformed BB (de Campos and Ji 2011). The first iteration of ARA* is equivalent to WA*, so we assume the results of that study extend here.

One objective in this study is to compare the anytime behavior of these algorithms, including the quality (i.e, score) of the anytime solution and the error bounds. The other objective is better understand the shortest-path formulation. To rigorously study both objectives, we generated test datasets by sampling synthetic Bayesian networks. For all experiments, we first selected a number of variables and maximum number of parents allowed for each variable. We then created the networks using a slight variation on the Ide-Cozman MCMC algorithm (Ide and Cozman 2002). During the MCMC process, if a successor network resulted in a variable exceeding the maximum number of parents, that network was discarded, and the MCMC continued from the previous network. All variables were binary, and conditional probability distributions were sampled from a symmetric Dirichlet distribution with a concentration parameter of 1. We call these networks the *generating networks*. Then, we generated datasets with 1,000 to 20,000 data points with logic sampling.

Dataset	POPS	Best score	60 Second Score				60 Second Error Bound				Final Score				Final Error Bound				Running Time (seconds)			
			AWinA*	ILP	ARA*	AWeiA*	AWinA*	ILP	ARA*	AWeiA*	AWinA*	ILP	ARA*	AWeiA*	AWinA*	ILP	ARA*	AWeiA*	AWinA*	ILP	ARA*	AWeiA*
29.3.1k	2164	**15298.15**	**1.000**	**1.000**	1.004	1.011	1.033	**1.000**	1.050	1.079	**1.000**	**1.000**	1.004	1.002	1.015	**1.000**	1.050	1.069	980*	45	998*	2140*
29.3.5k	20033	73927.94	**1.000**	1.007	1.012	1.015	1.065	**1.025**	1.100	1.113	**1.000**	**1.000**	1.004	1.006	1.033	**1.006**	1.050	1.102	1151*	7194	713*	219*
29.3.10k	47598	147625.66	**1.001**	1.011	1.014	1.008	1.072	**1.058**	1.124	1.118	**1.000**	1.003	1.008	1.008	1.039	**1.016**	1.100	1.118	1333*	7178	1587*	19*
29.3.20k	124564	293081.53	**1.002**	-	1.012	1.010	**1.093**	-	1.133	1.131	**1.000**	1.002	1.007	1.010	1.044	**1.022**	1.100	1.131	7261	7184	456*	2*
29.6.1k	1568	**15478.77**	**1.000**	**1.000**	1.003	1.015	1.014	**1.001**	1.041	1.060	**1.000**	**1.000**	**1.000**	1.005	**1.000**	1.001	**1.000**	1.050	279	6	290	181*
29.6.5k	16141	**72054.89**	**1.000**	1.002	1.003	1.010	1.030	**1.020**	1.050	1.081	**1.000**	**1.000**	**1.000**	1.001	**1.000**	**1.000**	**1.000**	1.071	283	6394	296	451*
29.6.10k	51542	**141306.05**	**1.000**	1.057	1.007	1.010	**1.061**	1.169	1.086	1.100	**1.000**	1.001	**1.000**	1.001	**1.000**	1.008	**1.000**	1.090	871	7200	1031	674*
29.6.20k	175340	**279387.44**	**1.000**	-	1.009	1.006	**1.077**	-	1.100	1.115	**1.000**	**1.000**	**1.000**	1.006	1.001	1.021	**1.000**	1.115	7208	7140	5127	59*
31.3.1k	1290	**16566.76**	1.001	**1.000**	1.009	1.010	1.053	**1.000**	1.082	1.083	**1.000**	**1.000**	1.004	1.010	1.027	**1.000**	1.050	1.083	982*	5	1161*	26*
31.3.5k	7885	80348.9	**1.002**	1.006	1.020	1.015	1.096	**1.022**	1.136	1.129	**1.000**	**1.000**	1.008	1.005	1.067	**1.004**	1.100	1.119	864*	7199	382*	463*
31.3.10k	20529	160304.6	**1.001**	1.007	1.026	1.006	1.114	**1.034**	1.159	1.138	**1.000**	1.003	1.016	1.006	1.084	**1.016**	1.141	1.138	938*	7181	1071*	44*
31.3.20k	52459	319897.03	**1.001**	1.016	1.020	1.010	**1.122**	1.139	1.162	1.151	**1.000**	1.003	1.014	1.010	1.099	**1.025**	1.150	1.151	1086*	7167	380*	4*
31.6.1k	1891	**18758.37**	1.007	**1.000**	1.017	1.013	1.030	**1.000**	1.046	1.042	**1.000**	**1.000**	1.017	1.003	1.002	**1.000**	1.038	1.032	1017*	23	877*	279*
31.6.5k	28299	**83224.24**	**1.000**	1.045	**1.000**	1.001	**1.000**	1.057	**1.000**	1.041	**1.000**	**1.000**	**1.000**	1.001	**1.000**	**1.000**	**1.000**	1.041	13	2922	14	14*
31.6.10k	98824	**161836.84**	**1.000**	-	**1.000**	1.001	**1.000**	-	1.017	1.052	**1.000**	1.001	**1.000**	1.001	**1.000**	1.010	**1.000**	1.052	42	7185	63	13*
31.6.20k	375404	**318807.50**	**1.000**	-	1.016	1.003	**1.041**	-	1.048	1.066	**1.000**	1.011	**1.000**	1.003	**1.000**	1.026	**1.000**	1.066	277	7193	334	37*
33.3.1k	2075	**17356.83**	1.006	**1.000**	1.019	1.022	1.072	**1.000**	1.100	1.111	1.001	**1.000**	1.019	1.012	1.040	**1.000**	1.096	1.101	910*	24	881*	1226*
33.3.5k	18491	83689.7	**1.006**	1.008	1.030	1.020	1.121	**1.039**	1.166	1.155	**1.000**	1.002	1.019	1.011	1.069	**1.013**	1.148	1.145	1111*	7182	70*	279*
33.3.10k	55083	167095.75	**1.004**	1.020	11.025	1.012	**1.131**	1.158	1.173	1.159	**1.000**	1.002	1.015	1.012	1.089	**1.022**	1.150	1.159	1308*	7126	4302*	23*
33.3.20k	184523	333144.78	**1.010**	-	1.015	1.014	**1.163**	-	1.179	1.178	**1.000**	1.002	1.015	1.006	1.104	**1.030**	1.179	1.168	7221	6976	39*	1093*
33.6.1k	2179	**19101.15**	1.008	**1.000**	1.021	1.015	1.025	**1.000**	1.045	1.041	**1.000**	**1.000**	1.021	1.005	1.001	**1.000**	1.034	1.031	1072*	49	949*	88*
33.6.5k	29393	**86040.53**	**1.000**	1.032	1.001	1.015	**1.001**	1.038	1.020	1.058	**1.000**	**1.000**	**1.000**	1.005	**1.000**	**1.000**	**1.000**	1.048	66	4073	110	80*
33.6.10k	93186	**168673.1**	**1.000**	-	1.015	1.013	**1.026**	-	1.050	1.066	**1.000**	1.001	**1.000**	1.003	**1.000**	1.005	**1.000**	1.055	243	7173	255	114*
33.6.20k	357832	331711.44	**1.000**	-	1.006	1.015	1.050	-	**1.046**	1.082	**1.000**	1.012	**1.000**	1.001	**1.000**	1.021	**1.000**	1.061	1397	6936	1425	2561*
35.3.1k	4321	**16935.32**	**1.003**	1.005	1.021	1.033	1.065	**1.010**	1.100	1.124	**1.000**	**1.000**	1.021	1.024	1.036	**1.000**	1.092	1.114	938*	84	749*	890*
35.3.5k	46429	81636.25	**1.005**	1.026	1.022	1.018	1.120	**1.109**	1.142	1.154	**1.000**	1.003	1.022	1.009	1.072	**1.019**	1.142	1.144	1307*	7123	24*	4234*
35.3.10k	151501	162883.4	**1.008**	-	1.039	1.021	**1.145**	-	1.197	1.176	1.003	**1.000**	1.020	1.012	1.097	**1.026**	1.150	1.166	7245	7021	2779*	237*
35.3.20k	462115	326013.53	**1.010**	-	1.022	1.017	**1.176**	-	1.187	1.191	**1.000**	**1.000**	1.022	1.009	1.117	**1.039**	1.187	1.181	7232	7185	57*	112*
35.6.1k	2111	**20536.60**	1.019	**1.000**	1.029	1.018	1.052	**1.000**	1.070	1.058	1.008	**1.000**	1.029	1.018	1.029	**1.000**	1.066	1.058	928*	40	572*	48*
35.6.5k	31608	**91850.43**	**1.000**	1.031	1.025	1.039	**1.022**	1.039	1.068	1.097	**1.000**	**1.000**	**1.000**	1.011	**1.000**	**1.000**	**1.000**	1.067	799	6087	1089	848*
35.6.10k	108155	**179521.58**	**1.000**	-	1.018	1.032	**1.050**	-	1.060	1.100	**1.000**	1.001	**1.000**	1.001	**1.000**	1.010	**1.000**	1.070	2024	7199	4993	2788*
35.6.20k	401515	353827.4	**1.005**	-	1.100	1.056	**1.080**	-	1.190	1.140	**1.000**	1.021	1.003	1.002	**1.010**	1.040	1.050	1.070	7203	4229	685*	5359*

Table 1: A comparison of the quality of 60-second and final solutions plus error bounds of the algorithms. A dataset named “a.b.ck” stands for one with a variables, maximum b parents, and ck data points. “POPS” shows the number of possibly optimal parent sets. “Best score” shows the MDL score of the best solution found by any of the algorithms; a boldfaced number indicates it was proven optimal. “60 Second Score” and “60 Second Error Bound” are the score (shown as the ratio over the best score) and error bound of solutions found by each algorithm after 60 seconds. A boldfaced number here indicates the best result among all algorithms. An error bound of “1.000” means a solution is proven optimal. “-” means that ILP did not find any solution within 60 seconds. Similarly, “Final Score” and “Final Error Bound” are for the solutions found after 2 hours. Finally, the running time until the last error bound improvement of the algorithms are reported in the column “Running Time”. The total runtimes are sometimes slightly longer than 2 hours because of discrepancies between wall time and CPU time. An asterisk in the running time means the algorithm exhausted RAM.

All of the algorithms we consider require a decomposable scoring function. In this evaluation, we use the MDL scoring function (Lam and Bacchus 1994). Let r_i be the number of states of X_i, N_{pa_i} be the number of data points consistent with $PA_i = pa_i$, and N_{x_i,pa_i} be the data points further constraint with $X_i = x_i$. Then MDL is given as follows.

$$MDL(G) = \sum_i MDL(X_i, PA_i), \tag{1}$$

where

$$\begin{aligned} MDL(X_i, PA_i) &= H(X_i, PA_i) + \frac{\log N}{2} K(X_i, PA_i), \\ H(X_i, PA_i) &= - \sum_{x_i, pa_i} N_{x_i,pa_i} \log \frac{N_{x_i,pa_i}}{N_{pa_i}}, \\ K(X_i, PA_i) &= (r_i - 1) \prod_{X_l \in PA_i} r_l. \end{aligned}$$

5.1 ANYTIME RESULTS

We first tested the anytime behavior of the algorithms on random networks with $\{29, 31, 33, 35\}$ variables and $\{3, 6\}$ maximum parents per variable. Then, from each network, we generated datasets with {1k, 5k, 10k, 20k} data points. Thus, in total, we considered 32 datasets. We put a 2-hour (7200 seconds) time limit on all the algorithms. The algorithms may terminate earlier than the time limit when either a provably optimal solution is found or RAM is exhausted. All of the algorithms (shortest-path-based and ILP) require the local scores ($MDL(X_i, PA_i)$) as input; therefore, we do not include these calculation times in the results. Table 1 shows all the results. We focus on synthetic networks in this study because we can control the parameters of the generating network; however, results from real-world data show similar trends.

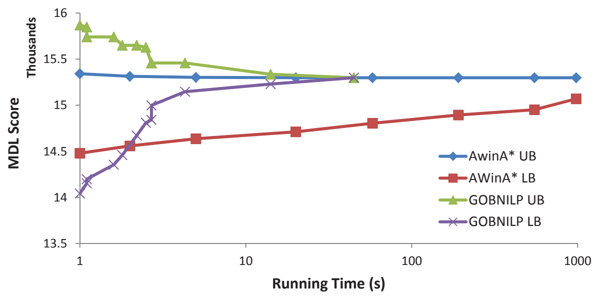
The results show that AWinA* performs much better than the other shortest-path-based algorithms. Its 60-second and final scores and error bounds are better than those of AWeiA* and ARA* on almost all cases. We note, however, that AWinA* often runs longer than the other algorithms before exhausting all the RAM. This is because AWinA* finds better solutions more quickly than AWeiA* and ARA*. Therefore, it prunes more nodes during the search and explores more of the search space. Consequently, it fills RAM more slowly and is able to run longer. For these reasons, we only consider AWinA* among the shortest-path-based algorithms for the remaining discussion.

ILP performed quite well on all the datasets with only 1k data points; it found all the optimal solutions within 60 seconds. AWinA* also often found the optimal solutions quickly, although it took much longer in proving the optimality, and sometimes failed to do so before running out of memory. The reason for ILP's excellent performance is that the numbers of possibly optimal parent sets (POPS) for these datasets are quite small. Therefore, the linear programs constructed by ILP are small and easy to solve.

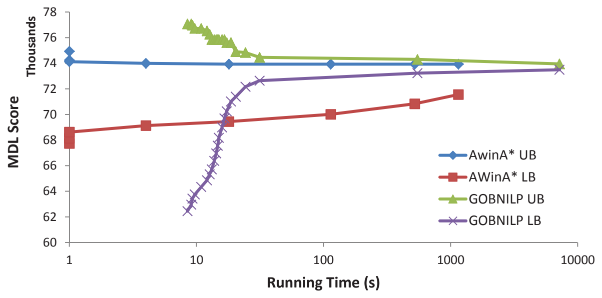
However, AWinA*'s 60-second and final solutions are all better than those found by ILP on the datasets with 5k, 10k and 20k data points, even though the error bound is sometimes worse than that of ILP. Those datasets had many more POPS, so each iteration of ILP required solving a large linear program. As a result, ILP was sometimes not able to find any solution within 60 seconds. The difference between scores at 60-seconds and the end of the search also show that ILP does not typically find its best solution early in the search. In contrast, AWinA* was always able to find a solution within several seconds. In fact, AWinA* found its best solution within the first 60 seconds on 14 of the datasets; the rest of the time is spent on proving the optimality of the solutions. This behavior is highly desirable. For a given large dataset, we do not know whether an optimal Bayesian network can be learned given limited resources. We should therefore strive to obtain as good a solution as we can as quickly as possible. The results show that AWinA* finds better solutions much sooner than ILP on many of the test datasets.

We note, however, ILP sometimes provides better error bounds than AWinA*. This is surprising given that its solutions are of lower quality, in terms of score, than those of AWinA*. The reason is that ILP often finds tighter lower bounds than AWinA*. The solutions of the LPs for ILP optimize the relaxed ILPs, so they give a lower bound on the solution. A simple local search is used to extract a valid BN from the LP solution and attempt to improve the upper bound. So most of the work in ILP focuses on improving the lower bound. On the other hand, the primary goal of AWinA* is to find a shortest path (subject to the sliding window constraint) from $start$ to $goal$, which improves the upper bound. Shortest-path-based algorithms must expand nodes with the lowest f-costs to improve the lower bounds, but the window semantics (and also weighted heuristic) discourages expanding nodes in early layers of the search, regardless of their f-cost. Therefore, AWinA* focuses more heavily on improving the upper bound.

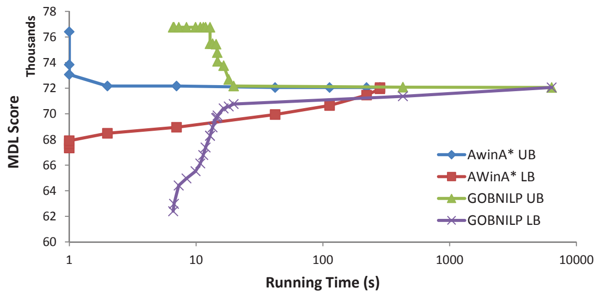
The better error bounds are certainly nice to have. AWinA* proved the optimality of its solutions for 13 of the datasets, while ILP proved optimality for 14. However, based on the error bounds of ILP, we can verify that AWinA* actually found optimal solutions for several other datasets. For example, for the "29.3.1k" dataset, both algorithms found a network with a score of $15,298.15$, but the final error bound for ILP is 1.00, while the bound for AWinA* is 1.07. Given ILP's error bound, then, we can conclude that AWinA* actually found the optimal network. Using this line of reasoning, we can see that AWinA* found the optimal network on 16 datasets, but ILP on only 14. The results suggest that ILP always either found-and-proved or did-not-find the optimal network on these datasets.



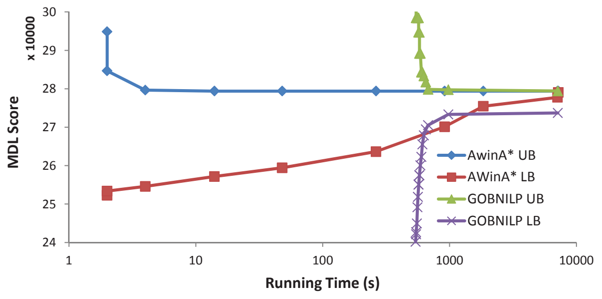
$(a)Dataset\ 29.3.1k$



$(b)Dataset\ 29.3.5k$



$(c)Dataset\ 29.6.5k$



$(b)Dataset\ 29.6.20k$

Figure 2: A comparison of the convergence of upper bounds (UB) and lower bounds (LB) for AWinA* and ILP.

Another difference between the two algorithms is that AWinA* often terminates before the time limit because it exhausts all of the available RAM storing *open* and *closed* lists. On the other hand, ILP typically runs out of time, but does not fully utilize RAM. These results suggest that ILP may be able to find solutions of the same quality as AWinA* if given enough time, but it is unclear how much more time would be needed. Similarly, more RAM or an external-memory strategy could improve the quality and error bounds of the solutions of AWinA*.

5.2 CONVERGENCE OF BOUNDS

To gain a better perspective on how AWinA* and ILP improve the upper and lower bounds, we plot the convergence curves of the bounds against the running time for several datasets in Figure 2. The results clearly agree with our analysis in Section 5.1. AWinA* was able to find good solutions very quickly, while ILP was slower to find its first solution. Also, even though ILP finds quite bad lower bounds initially, it quickly improves them. Finally, the pace with which AWinA* improves its solutions and error bounds is quite regular (increasing roughly exponentially from one iteration to the next). In comparison, ILP was able to improve its solution quickly and often in the early stage of the search, but its pace slowed down significantly in the later stages. This suggests that ILP may need much longer to find the next solution. For the "29.3.1k" dataset, ILP found and proved the optimal network in 45 seconds. Even though AWinA* initially found better solutions than ILP, and the optimal solution in 58 seconds, it failed to prove its optimality before running out of memory. For the "29.3.5k" dataset, both algorithms failed to prove the optimality of their solutions. AWinA* found its best solution in 18 seconds. Based on its behavior on other datasets, we suspect that AWinA* found the optimal solution but ran out of RAM before proving its optimality. ILP's solution was worse than that of AWinA*, so it definitely did not find the optimal solution; however, it obtained a much better lower bound and, hence, a better error bound. For the "29.6.5k" dataset, both algorithms were able to prove optimality of the solutions. AWinA* was able to find the optimal solution in 42 seconds and prove it in 283 seconds, while ILP only finds the optimal solution near the end of the search (6,394s). Finally for the "29.6.20k" dataset, AWinA* found the optimal solution in 14 seconds and proved its optimality close to the time limit. ILP took much longer (497s) before finding its first solution, and was not able to find the optimal solution within the time limit.

5.3 EFFECT OF GENERATING PARAMETERS

We created the generating networks by varying the numbers of variables and maximum parents allowed for each variable, and generated the testing datasets with different numbers of data points. In general, more variables or more data points makes a dataset more difficult to solve optimally, and, hence, increased the error bounds of both AWinA* and ILP. Relatively, the number of data points has a larger effect on ILP; it solved almost all 1k datasets and several 5k datasets optimally, but none of the 10k or 20k datasets. The reason is that more data points increase the

Dataset	Percentage	Generating	Learned	Distance
29.2.1k	41.92	1.72	1.62	7
29.2.5k	80.16	1.72	1.69	3
29.2.10k	89.40	1.72	1.72	0
29.2.20k	94.37	1.72	1.72	0
29.4.1k	1.79	3.03	3	1
29.4.5k	8.81	3.03	3.03	0
29.4.10k	18.54	3.03	3.03	0
29.4.20k	34.35	3.03	3.03	0
29.6.1k	0.22	4.24	3.45	23
29.6.5k	0.03	4.24	4.24	3
29.6.10k	0.11	4.24	4.21	1
29.6.20k	0.44	4.24	4.21	1
29.8.1k	18.59	5.03	2.52	95
29.8.5k	0.48	5.03	4.66	11
29.8.10k	0.23	5.03	4.97	9
29.8.20k	0.20	5.03	5.03	0

Table 2: The percentage of search nodes with better f-cost than the optimal solution ("Percent"); the average number of parents in the original ("Original") and learned networks ("Learned"), and the structural Hamming distance between the original and learned networks ("Distance").

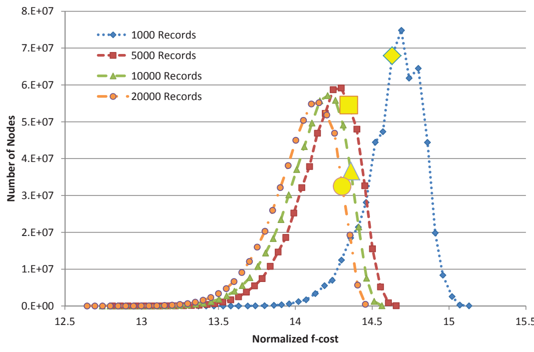
number of POPS and make the linear programs larger and harder to solve.

A somewhat surprising observation comes from the effect of the maximum allowed parents in the generating networks on the the algorithms. AWinA* was able to solve almost all the datasets that allow up to 6 parents (6-parent datasets for short hereafter) optimally, but none of the 3-parent datasets. Similarly, ILP was able to find optimal solutions for more 6-parent datasets than 3-parent datasets. This is surprising because, intuitively, more parents make the Bayesian networks more complex and seemingly harder to learn. To better understand the effect of the generating parameters on the algorithms, especially AWinA*, we performed a more detailed sensitivity analysis.

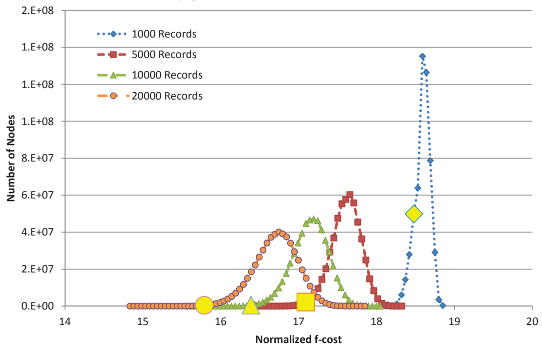
5.4 SENSITIVITY ANALYSIS OF PARAMETERS

To study the sensitivity of shortest-path-based algorithms to the parameters for network and dataset generation, we generated networks with: 29 variables and {2, 4, 6, 8} maximum parents per variable. Then, from each network, we again generated datasets with: {1k, 5k, 10k, 20k} data points. We take the number of parameters necessary to specify the conditional probability distributions in the generating network as a measure of complexity (i.e., more parameters mean a more complex process).

For each dataset, we first collected the f-costs of all the nodes in the order graph for each dataset using a BFS search. Because we were interested only in the characteristics of the search space, we did not impose any time limit on the algorithm; it can effectively use external memory, so that resource did not pose a problem, either. Table 2 shows the percentages of search nodes that have better f-costs than the optimal solution, as well as several other statistics. We



(a) Datasets 29.2.*



(b) Datasets 29.8.*

Figure 3: Distributions of the f-costs of all the search nodes normalized by the number of data points. The enlarged marks indicate where the optimal solutions are located.

also show distributions of the f-costs of all nodes relative to the optimal solutions in Figure 3.

As shown in Equation 1, the MDL score consists of two terms: the log-likelihood of the data given the structure and a structure complexity penalty. The likelihood term increases linearly in the number of data points, while the term that penalizes structural complexity increases logrithmically in the number of data points. Figure 4 shows that when the number of data points is small, possibly optimal parent sets (POPS) are typically smaller; consequently, learned optimal networks tend to be simpler than generating networks. As the number of data points increases, POPS become larger and learned networks more complex.

Indeed, as Table 2 shows, the average number of parents in the learned, optimal network increases as the number of data points increases (except for a slight decrease from 5k to 10k data points for the 6-parent networks). As the number of data points increases, the structural Hamming distance between the learned, optimal network and the generating network decreases and drops to 0 for several datasets. This shows that, given enough data, MDL can recover the generating network and is appropriate for study. In addition, Figure 3 shows that the normalized f-costs of the optimal solutions shift left with increasing data points; this is because more variables used larger parent sets and obtained better scores. We also observe that the f-cost distributions of all search nodes shifted leftward. Because of the heuristic functions used in A*, the internal order graph nodes relax the acyclic constraints between some of the variables, and have even more freedom to use the larger POPS. As a result, more internal nodes obtained better f-costs.

Therefore, the percentages of f-costs better than the optimal solutions depend on the relative speed in which the optimal solutions and f-cost distributions shift. For the 8-parent datasets, when we have few data points, even though the generating network is complex, the relatively large complexity penalty forces the search to consider simple networks which do not explain the data as well but incur a small complexity penalty. As Table 2 shows, based on the average number of parents in the generating network compared to the optimal network for 1k records, the optimal network is quite a bit simpler than the generating network. As we add more data points, though, the likelihood term dominates the score calculations. Therefore, the complex structures which better explain the data have, relatively, much better scores than simpler structures. Figure 3(b) shows that the optimal solutions shift to the left relative to the other nodes for the 8-parent datasets as the number of data points increases. That explains why the percentage of nodes with better f-costs than the optimal network is high, but decreases as the number of data points increases.

It is a different story for the 2- and 4-parent datasets. As Table 2 shows, for the 2-parent datasets, the percentages of nodes with f-costs better than the optimal network are rather high. The percentages increase with the number of data points and approach 95% for the 20k dataset. For the 4-parent datasets, the percentage is initially low but increases significantly as the number of data points increases. To understand why, we again consult Table 2, which shows that the generating networks for those datasets do not have many parents for each node. Therefore, simpler structures both explain the data well and have a low complexity penalty. Unlike in the 8-parent case, more complex structures can not improve upon the likelihood very much but incur a much larger complexity penalty. Consequently, fewer data points are needed to predict the structures well. The results show that even with only 1k data points, the learned networks have similar numbers of parents and structures as the generating networks. So the learned, optimal networks have converged to the generating networks and do not benefit much from more data points. Figure 3(a) indeed shows that the optimal solutions did not shift left much with more than 5k data points; they actually shift towards the right tails of the distributions with more data points.

These results help explain the performance, particularly the error bounds, of AWinA* in the first set of experiments. To completely prove optimality, AWinA* would have to expand all nodes with better f-costs than the goal. In practice, though, it can only expand about 10 million nodes in a search space in the allocated resources. These results suggest that AWinA* would not be able to prove optimality for datasets generated from simple networks and a large number of data points. Table 1 shows this is exactly the case.

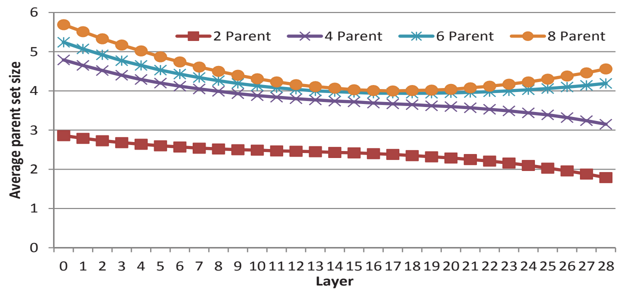


Figure 4: Average parent set size of all the nodes in each layer of the order graph for the “29.*.5k” datasets.

As further evidence, Figure 4 shows the average parent set size of the (cyclic) networks corresponding to all search nodes in each layer of the order graph for the “29.*.5k” datasets. For the 2- and 4-parent datasets, the average cardinality decreases monotonically. The heuristic estimates of most search nodes seem to select larger parent sets and have lower costs than the goal, so they would have to be expanded by A*. For the 6- and 8-parent datasets, the average cardinality dips initially and then increases. Many nodes in the middle layers seem to select smaller parent sets and have higher costs than the goal, so many of them are never expanded. The rate of the change of average cardinality is often larger in the beginning and last layers. The explanation is that the beginning and last layers have much fewer search nodes than the middle layers, so the changes in parent sets have a larger effect on the average cardinality.

6 CONCLUSIONS

In this research, we adapted several anytime heuristic search algorithms to learn optimal Bayesian networks from data, and empirically evaluated these algorithms against an integer linear programming algorithm. Our empirical results show that AWinA* is the best-performing anytime algorithm among the ones we evaluated in this study. It finds better, often optimal, solutions more quickly than existing methods; in many cases, the majority of its running time is spent on proving the optimality of a solution found early in the search. In comparison, the ILP algorithm focuses on finding lower bounds for the optimal solution in its search. As a result, its lower bounds are often better than those of AWinA*, even though its solutions are often not as good.

Our results show that, surprisingly, complex generating networks may seem structurally challenging to learn, but they actually lie within the promising solution space that is first explored by heuristic search and are easy to find.

Acknowledgements This work was supported by NSF grants IIS-0953723 and IIS-1219114.

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 Artifical intelligence*, IJCAI'07, 2250–2255. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Buntine, W. 1991. Theory refinement on Bayesian networks. In *Proceedings of the seventh conference (1991) on Uncertainty in artificial intelligence*, 52–60. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Chickering, D. M. 1996. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V*, 121–130. Springer-Verlag.

Chickering, D. M. 2002. Learning equivalence classes of Bayesian-network structures. *J. Mach. Learn. Res.* 2.

Cussens, J. 2011. Bayesian network learning with cutting planes. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI-11)*, 153–160. Corvallis, Oregon: AUAI Press.

de Campos, C. P., and Ji, Q. 2011. Efficient learning of Bayesian networks using constraints. *Journal of Machine Learning Research* 12:663–689.

Hansen, E. A., and Zhou, R. 2007. Anytime heuristic search. *Journal of Artificial Intelligence Research* 28.

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. 1996. A tutorial on learning with Bayesian networks. Technical report, Learning in Graphical Models.

Ide, J. S., and Cozman, F. G. 2002. Random generation of bayesian networks. In *Brazillian Symposium on Artificial Intelligence*, 366–375.

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 (AISTATS)*.

Koivisto, M., and Sood, K. 2004. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research* 549–573.

Lam, W., and Bacchus, F. 1994. Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence* 10:269–293.

Likhachev, M.; Gordon, G.; and Thrun, S. 2003. ARA*: Anytime A* search with provable bounds on sub-optimality. In Thrun, S.; Saul, L.; and Schölkopf, B., eds., *Proceedings of Conference on Neural Information Processing Systems (NIPS)*. MIT Press.

Malone, B., and Yuan, C. 2012. A parallel, anytime, bounded error algorithm for exact bayesian network structure learning. In *Proceedings of the Sixth European Workshop on Probabilistic Graphical Models (PGM-12)*.

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 *Conference on Uncertainty in Artificial Intelligence (UAI-11)*, 479–488. Corvallis, Oregon: AUAI Press.

Malone, B.; Yuan, C.; and Hansen, E. 2011. Memory-efficient dynamic programming for learning optimal Bayesian networks. In *National conference on Artifical intelligence*.

Moore, A., and Wong, W.-K. 2003. Optimal reinsertion: A new search operator for accelerated and more accurate Bayesian network structure learning. In *Intl. Conf. on Machine Learning*, 552–559.

Ott, S.; Imoto, S.; and Miyano, S. 2004. Finding optimal models for small gene networks. In *Pac. Symp. Biocomput*, 557–567.

Pohl, I. 1970. Heuristic search viewed as path finding in a graph. *Artificial Intelligence* 1(3-4):193 – 204.

Silander, T., and Myllymaki, P. 2006. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*. Arlington, Virginia: AUAI Press.

Silander, T.; Roos, T.; Kontkanen, P.; and Myllymaki, P. 2008. Factorized normalized maximum likelihood criterion for learning Bayesian network structures. In *Proceedings of the 4th European Workshop on Probabilistic Graphical Models (PGM-08)*, 257–272.

Singh, A., and Moore, A. 2005. Finding optimal Bayesian networks by dynamic programming. Technical report, Carnegie Mellon University.

Teyssier, M., and Koller, D. 2005. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of the Twenty-First Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*, 584–590. Arlington, Virginia: AUAI Press.

Yuan, C., and Malone, B. 2012. An improved admissible heuristic for finding optimal bayesian networks. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence (UAI-12)*. AUAI Press.

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*.

On the Complexity of Strong and Epistemic Credal Networks

Denis D. Mauá **Cassio P. de Campos** **Alessio Benavoli** **Alessandro Antonucci**
Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA),
Manno-Lugano, Switzerland,
{denis,cassio,alessio,alessandro}@idsia.ch

Abstract

Credal networks are graph-based statistical models whose parameters take values in a set, instead of being sharply specified as in traditional statistical models (e.g., Bayesian networks). The computational complexity of inferences on such models depends on the irrelevance/independence concept adopted. In this paper, we study inferential complexity under the concepts of epistemic irrelevance and strong independence. We show that inferences under strong independence are NP-hard even in trees with ternary variables. We prove that under epistemic irrelevance the polynomial time complexity of inferences in credal trees is not likely to extend to more general models (e.g. singly connected networks). These results clearly distinguish networks that admit efficient inferences and those where inferences are most likely hard, and settle several open questions regarding computational complexity.

1 INTRODUCTION

Bayesian networks are multivariate statistical models where irrelevance assessments between sets of variables are concisely described by means of an *acyclic directed graph* whose nodes are identified with variables (Pearl, 1988). The graph encodes a set of *Markov conditions*: non-descendant non-parent variables are irrelevant to a variable given its parents. The complete specification of a Bayesian network requires the specification of a conditional probability distribution for every variable and every configuration of its parents. This is no easy task. Whether these parameters (i.e., the conditional distributions) are estimated from data or elicited from experts, they inevitably contain imprecisions and arbitrariness (Kwisthout and van der Gaag, 2008). Despite this fact, Bayesian networks have been successfully applied to a wide range of applications (Koller and Friedman, 2009).

An arguably more principled approach to coping with imprecision in parameters is by means of closed and convex sets of probability mass functions called *credal sets* (Levi, 1980). Unlike the representation of knowledge by "precise" probability mass functions, credal sets allow for the distinction between randomness and ignorance (Walley, 1991). Bayesian networks whose parameters are specified by conditional credal sets are known as *credal networks* (Cozman, 2000). Credal networks have been successfully applied to knowledge-based expert systems, where it has been argued that allowing parameters to be imprecisely specified facilitates elicitation from experts (Antonucci et al., 2007, 2009; Piatti et al., 2010).

A Bayesian network provides a concise representation of the (single) multivariate probability mass function that is consistent with the network parameters and factorizes over the graph. Analogously, a credal network provides a concise representation of the credal set of multivariate mass functions that are consistent with the local credal sets and satisfy (at least) the irrelevances encoded in the graph. The precise characterization of this joint credal set depends however on the concept of irrelevance adopted.

The two most commonly used irrelevance concepts in the literature are *strong independence* and *epistemic irrelevance*. Two variables X and Y are *strongly independent* if the joint credal set of X, Y can be regarded as originating from a number of precise probability mass functions in each of which the two variables are stochastically independent. Strong independence is thus closely related to the sensitivity analysis interpretation of credal sets, which regards an imprecisely specified model as arising out of partial ignorance of an ideal precisely specified one (Kwisthout and van der Gaag, 2008; Antonucci and Piatti, 2009; Zaffalon and Miranda, 2009). A variable X is *epistemically irrel-*

evant to a variable Y if the marginal credal set of Y according to our model is the same whether we observe the value of X or not (Walley, 1991).

Typically, credal networks are used to derive tight bounds on the expectation of some variable conditional on the value of some other variables. The complexity of such an inferential task varies greatly according to the topology of the underlying graph, the cardinality of the variable domains, and the irrelevance concept adopted. For instance, the 2U algorithm of Fagiuoli and Zaffalon (1998) can solve the problem in polynomial time if the underlying graph is singly connected, variables are binary and strong independence is assumed. When instead epistemic irrelevance is adopted, no analogous polynomial-time algorithm for the task is known. On the other hand, de Cooman et al. (2010) developed a polynomial-time algorithm for inferences in credal trees under epistemic irrelevance that work for arbitrarily large variable domains. No such algorithm is known under strong independence. Recently, Mauá et al. (2012) showed the existence of a fully polynomial-time approximation scheme for credal networks of bounded treewidth and bounded variable cardinality under strong independence. It is still unknown whether an analogous result can be obtained under epistemic irrelevance.

In this paper, we show that epistemic irrelevance and strong independence induce the same upper and lower predictive probability in HMM-like (hidden Markov model-like) credal networks (i.e., when we query the value of the "last" state node given some evidence), and that they induce the same marginal expectation bounds in any network where non-root nodes are precise and root nodes are vacuous. The former implies that we can use the algorithm of de Cooman et al. (2010) for epistemic credal trees to compute tight bounds on the predictive probability in HMM-like credal networks also under strong independence. The latter implies that computing tight posterior probability bounds in singly connected credal networks over ternary variables under epistemic irrelevance is NP-hard, as we show this to be the case under strong independence (de Campos and Cozman (2005) have previously shown the NP-hardness of inference in singly connected strong networks when variable cardinalities are unbounded). Table 1 summarizes both the previously known complexity results and the contributions of this work, which appear in boldface.

Table 1: Inferential Complexity of Credal Networks

MODEL	**STRONG**	**EPISTEMIC**
(Predictive) HMM	**P**	P
Tree	**NP-hard**	P
Singly connected	**NP-hard**	**NP-hard**
Multiply connected	NP^{PP}-hard	$\mathbf{NP^{PP}}$**-hard**

2 CREDAL NETWORKS

Let $X_1, \ldots, X_n$ be variables taking values $x_1, \ldots, x_n$ in finite sets $\mathcal{X}_1, \ldots, \mathcal{X}_n$, respectively. We write $X := (X_1, \ldots, X_n)$ to denote the n-dimensional variable taking values x in $\mathcal{X} := \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$. Given X and $I \subseteq \{1, \ldots, n\} := N$, the notation X_I denotes the vector obtained from X by discarding the coordinates not in I. We also write $\mathcal{X}_I$ to denote the Cartesian product (in the proper order) of the sets $\mathcal{X}_i$ with $i \in I$, and x_I to denote an element of $\mathcal{X}_I$. Note that $X_N = X$ and $\mathcal{X}_N = \mathcal{X}$.

Let $\mathcal{Z}$ be some finite set (multidimensional or not), and Z a variable taking values in $\mathcal{Z}$. A *probability mass function (pmf)* $p(Z)$ is a non-negative real-valued function on $\mathcal{Z}$ such that $\sum_{z \in \mathcal{Z}} p(z) = 1$. Given a pmf p on $\mathcal{Z}$, we define the *expectation* operator as the functional $\mathbb{E}_p[f] := \sum_{z \in \mathcal{Z}} f(z)p(z)$ that maps every real-valued function f on $\mathcal{Z}$ to a real number. Given a pmf $p(X)$ on $\mathcal{X}$, and disjoint subsets I, J and K, we say that variables X_K are *stochastically irrelevant* to X_I given X_J if $p(x_I|x_{J \cup K}) = p(x_I|x_J)$ for all x, where the conditional pmfs are obtained by application of Bayes' rule on $p(X)$. Variables X_I and X_K are independent conditional on X_J if, given X_J, X_I and X_K are irrelevant to each other.

A *credal set* $C(Z)$ is a closed and convex set of pmfs $p(Z)$ on $\mathcal{Z}$ (Levi, 1980). The *extrema* of a credal set are the points that cannot be written as convex combinations of other points in the set. The extrema of $C(Z)$ are denoted by $\operatorname{ext} C(Z)$. We assume that every credal set has finitely many extrema, which are used to represent it. Thus, the credal sets we consider are geometrically polytopes. The *vacuous credal set* is the largest credal set on a given domain $\mathcal{Z}$, and is denoted by $V(Z)$. Given a set $M(Z)$ of pmfs $p(Z)$ on $\mathcal{Z}$ we write $\operatorname{co} M(Z)$ to denote its convex closure, that is, the credal set obtained by all convex combinations of elements of $M(Z)$. Given a credal set $C(X)$, disjoint subsets I and J of N, and an assignment x_J to X_J, we define the *conditional credal set* $C(X_I|x_J)$ (induced by $C(X)$) as the set $\operatorname{co}\{\, p(X_I|x_J) : p \in C(X),\, p(x_J) > 0\,\}$. It can be shown that the $C(X_I|x_J)$ remains the same if we replace $C(X)$ with its extrema $\operatorname{ext} C(X)$.

For disjoint subsets I, J and K of N, we say that a variable X_K is *strongly irrelevant* to X_I given X_J (and w.r.t. $C(X)$) if X_K is stochastically irrelevant to X_I given X_J in every extreme $p \in \operatorname{ext} C(X)$. This implies that $C(X_I|x_{J \cup K}) = C(X_I|x_J)$ for all x. Variables X_I and X_K are *strongly independent* given X_J if condi-

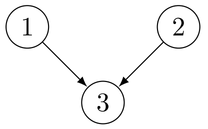


Figure 1: A Simple Polytree

tional on X_J both X_I is strongly irrelevant to X_K and X_K is strongly irrelevant to X_I. Since stochastic irrelevance implies stochastic independence, strong independence is implied by strong irrelevance.

We say that variables X_K are *epistemically irrelevant* to X_I given X_J if $C(X_I|x_{J\cup K}) = C(X_I|x_J)$ for all values of x. One can show that strong irrelevance implies epistemic irrelevance (and the converse is not necessarily true) (Cozman, 2000; de Cooman and Troffaes, 2004). Variables X_I and X_K are *epistemically independent* conditional on X_J if, given X_J, X_I and X_K are epistemically irrelevant to each other (Walley, 1991, Ch. 9).

Let $G = (N, A)$ be an acyclic directed graph. We denote the *parents* of a node i by $\mathrm{pa}(i)$. The set of *non-descendants* of i, written $\mathrm{nd}(i)$, contains the nodes not reachable from i by a directed path. Note that $\mathrm{pa}(i) \subseteq \mathrm{nd}(i)$. We say that G is *singly connected* if there is at most one *undirected* path connecting any two nodes in the graph; it is a *tree* is if additionally each node has at most one parent. If a graph is not singly connected, we say it is *multiply connected.* Singly connected directed graphs are also called *polytrees.*

A (separately specified) credal network $\mathcal{N}$ associates to each node i in G a variable X_i and a collection $Q(X_i|X_{\mathrm{pa}(i)})$ of local credal sets $Q(X_i|x_{\mathrm{pa}(i)})$ indexed by the values of $X_{\mathrm{pa}(i)}$. When every local credal set is a singleton the model specifies a Bayesian network.

Example 1. *Consider the credal network $\mathcal{N}$ over Boolean variables X_1, X_2, X_3 whose graph is shown in Figure 1. Let $[p(0), p(1)]$ represent a pmf of a Boolean variable. The local credal sets are $Q(X_1) = Q(X_2) = \mathrm{co}\{[0.4, 0.6], [0.5, 0.5]\}$ and $Q(X_3|x_{1,2}) = \{[I(x_1 = x_2), I(x_1 \neq x_2)]\}$, where $I(\cdot)$ is the indicator function.*

The *strong extension* is the credal set $C(X)$ whose extrema $p(X)$ satisfy for all x the condition

$$p(x) = \prod_{i\in N} q(x_i|x_{\mathrm{pa}(i)}), \qquad (1)$$

where $q(X_i|x_{\mathrm{pa}(i)}) \in \mathrm{ext}\, Q(X_i|x_{\mathrm{pa}(i)})$. The strong extension satisfies the Markov condition w.r.t. strong independence: every variable is strongly independent of its non-descendant non-parents given its parents. The *epistemic extension* is the joint credal set $C(X)$ such that

$$C(X_i|x_{\mathrm{nd}(i)}) = Q(X_i|x_{\mathrm{pa}(i)}) \qquad (2)$$

for every variable X_i and value $x_{\mathrm{nd}(i)}$ of $X_{\mathrm{nd}(i)}$. The epistemic extension satisfies the Markov condition w.r.t. epistemic irrelevance: the non-descendant non-parents are irrelevant to a variable given its parents. Equation 2 implies that (and is equivalent to for Boolean variables X_i)

$$\min q(x_i|x_{\mathrm{pa}(i)}) \leq p(x_i|x_{\mathrm{nd}(i)}) \leq \max q(x_i|x_{\mathrm{pa}(i)}) \qquad (3)$$

for all x and $p \in C(X)$ with $p(x_{\mathrm{nd}(i)}) > 0$, where the optimizations are over $q \in Q(X_i|x_{\mathrm{pa}(i)})$. Note that these inequalities can be turned into linear inequalities by multiplying both sides by $p(x_{\mathrm{nd}(i)})$.

Given a function f of a query variable X_q, and an assignment $\tilde{x}_O$ to evidence variables X_O, the primary inference with credal networks is the application of the *generalized Bayes rule* (GBR), which asks for a value of μ that solves the equation

$$\min_{p\in C(X)} \sum_{x\sim\tilde{x}_O} [f(x_q) - \mu]p(x) = 0\,, \qquad (4)$$

where the sum is performed over the values x of X whose coordinates indexed by O equal $\tilde{x}_O$. Assuming that $\min_{p\in C(X_O)} p(\tilde{x}_O) > 0$, it follows that

$$\mu = \min_{p\in C(X_q|\tilde{x}_O)} \mathbb{E}_p[f]\,, \qquad (5)$$

that is, μ is the lower expectation of f on the posterior credal set $C(X_q|\tilde{x}_O)$ induced by the (strong or epistemic) extension of the network.

Example 2. *Consider the network in Example 1, and let $q = 3$, $f = I(x_3 = 0)$ and O is the empty set. Then applying the GBR is equivalent to finding the lower marginal probability $\mu = \min_{p\in C(X_3)} p(0)$ induced by the network extension. Assuming strong independence (hence strong extension), the inference is solved by $\mu = \min \sum_{x_{1,2}} q_1(x_1)q_2(x_2)q_3(0|x_{1,2}) = 1 + \min\{2q_1(0)q_2(0) - q_1(0) - q_2(0)\} = 1 - 1/2 = 1/2\,$, where the minimizations are performed over $q_1(X_1) \in \mathrm{ext}\, Q(X_1)$, $q_2(X_2) \in \mathrm{ext}\, Q(X_2)$, and $q_3(X_3|x_{1,2}) = [I(x_1 = x_2), I(x_1 \neq x_2)]$. The epistemic extension is the credal set of joint pmfs $p(X)$ on $\mathcal{X}$ such that*

$$\min_{q\in Q(X_1)} q(x_1) \leq p(x_1|x_2) \leq \max_{q\in Q(X_1)} q(x_1)\,,$$
$$\min_{q\in Q(X_1)} q(x_2) \leq p(x_2|x_1) \leq \max_{q\in Q(X_1)} q(x_2)\,,$$
$$q(x_3|x_{1,2}) \leq p(x_3|x_{\{1,2\}}) \leq q(x_3|x_{1,3})\,.$$

The inference under epistemic irrelevance is the value of the solution of the linear program $\mu = \min\{p(0,0,0) + p(1,1,0) : p(X) \in C(X)\} = 5/11 < 1/2$, where $C(X)$ is the epistemic extension.

3 COMPLEXITY RESULTS

In this section we present new results about the computational complexity of GBR inferences in credal networks. We make use of previously unknown equivalences between strong and epistemic extensions to derive both positive and negative complexity results. The section is divided in subsections addressing networks in increasing order of topological complexity.

3.1 HIDDEN MARKOV MODELS

An imprecise hidden Markov model (HMM) is a credal network whose nodes can be partitioned into *state* and *manifest* nodes such that the state nodes form a chain (i.e., a sequence of nodes with one node linking to the next and to no other in the sequence), and each manifest node is a leaf with a single state node as parent. As the following example shows, there are GBR inferences in HMMs which depend on the irrelevance concept adopted, even in the case of binary variables.

Example 3. *Consider an HMM over four Boolean variables X_1, X_2, X_3, X_4, where X_1 and X_2 are state variables and X_3 and X_4 are manifest variables. The topology of the corresponding credal network is depicted in Figure 2. The local credal sets are given by $Q(X_1) = Q(X_2|0) = Q(X_4|0) = \{[3/4, 1/4]\}$, $Q(X_2|1) = Q(X_4|1) = \{[1/4, 3/4]\}$, and $Q(X_3|0) = \text{co}\{[1/4, 3/4], [1/2, 1/2]\}$ and $Q(X_3|1) = \text{co}\{[3/4, 1/4], [1/2, 1/2]\}$. Consider the query $f(X_4) = I(x_4 = 0)$, and evidence $\tilde{x}_3 = 0$. Under strong independence, the GBR is to solve for μ the equation*

$$\min \sum_{x_2} q(0|x_2) g_\mu(x_2) = \sum_{x_2} \min q(0|x_2) g_\mu(x_2) = 0\,,$$

where the minimizations are performed over $q(X_3|x_2) \in Q(X_3|x_2)$, $x_2 = 0, 1$, and

$$g_\mu(x_2) = \sum_{x_{1,4}} q(x_1) q(x_2|x_1) q(x_4|x_1) [I(x_4 = 0) - \mu]\,,$$

with $q(X_1) = q(X_2|0) = q(X_4|0) = [3/4, 1/4]$ and $q(X_2|1) = q(X_4|1) = [1/4, 3/4]$. The values of $q(0|x_2)$ depend only on the signs of $g_\mu(x_2)$, $x_2 = 0, 1$. Solving for μ for each of the four possibilities, and taking the minimum value of μ, we find that $\mu = \min\{p(0|0) : p(X_4|X_3) \in C(X_4|X_3)\} = 4/7$.

Under epistemic irrelevance, the GBR is equal to

$$\min \sum_{x_{1,2,4}} q(x_1) q(x_2|x_1) q(x_4|x_1) q(0|x_{1,2,4}) h_\mu(x_4) =$$

$$(1 - \mu) \sum_{x_{1,2}} q(x_1) q(x_2|x_1) q(0|x_1) \min q(0|x_{1,2}, x_4 = 0)$$

$$-\mu \sum_{x_{1,2}} q(x_1) q(x_2|x_1) q(1|x_1) \max q(0|x_{1,2}, x_4 = 1) = 0\,,$$

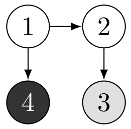


Figure 2: HMM Over Four Variables

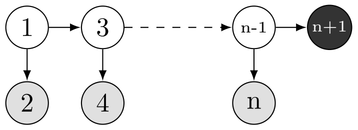


Figure 3: HMM Over $2n + 1$ Variables

where $h_\mu(x_4) = I(x_4 = 0) - \mu$, $q(X_1)$, $q(X_2|X_1)$ and $q(X_4|X_1)$ are defined as before, and $q(X_3|x_{1,2,4}) \in Q(X_3|x_2)$ for every $x_{1,2,4}$. Solving the equation above for μ we get that $\mu = 13/28$.

GBR inferences in HMMs are polynomial-time computable under epistemic irrelevance, but no polynomial-time algorithm is known under strong independence except for the case of binary variables (in which case the aforementioned 2U algorithm can be used). The following result shows that there exists a class of GBR inferences in HMMs which are insensitive to the irrelevance concept adopted. This implies that the GBR is polynomial-time computable in such cases also under strong independence.

Theorem 1. *Consider an HMM over $n + 1$ variables whose state nodes are identified with odd numbers and manifest nodes are identified with even numbers (see Figure 3). Assume that the query node is $q = n + 1$, and that evidence $\tilde{x}_O$ is set on a subset O of the manifest nodes. Then the posterior lower expectation of any function f on $\mathcal{X}_q$ conditional on $\tilde{x}_O$ is the same whether we assume epistemic irrelevance or strong independence.*

Proof. Let $g_\mu(x_{n+1})$ be equal to $f(x_{n+1}) - \mu$. To compute the GBR under strong independence, we need to find a number μ such that

$$\min \sum_{x \sim \tilde{x}_O} q(x_1) \prod_{i=2}^{n+1} q(x_i|x_{\text{pa}(i)}) g_\mu(x_{n+1}) = 0\,,$$

where the minimization is performed over $q(x_1) \in Q(X_1)$ and $q(X_i|x_{\text{pa}(i)}) \in Q(X_i|x_{\text{pa}(i)})$ for $i = 2, \ldots, n + 1$. The minimization above is equivalent to the constrained program

$$\text{minimize} \quad \sum_{x \sim \tilde{x}_O} q(x_1) \prod_{i=2}^{n+1} q(x_i|x_{1:i-1}) g_\mu(x_{n+1}) \qquad (6)$$

$$\text{subject to} \quad q(x_i|x_{1:i-1}) = q(x_i|x_{\text{pa}(i)}),\ i > 2\,, \qquad (7)$$

with variables $q(x_1) \in Q(X_1)$, $q(X_i|x_{\text{pa}(i)}) \in Q(X_i|x_{\text{pa}(i)})$, $i = 2,\ldots,n+1$ and $q(X_i|x_{1:i-1}) \in Q(X_i|x_{\text{pa}(i)})$, $i = 3,\ldots,n+1$. The objective function in (6) can be rewritten as

$$\sum_{x_{1:n}\sim\tilde{x}_O} q(x_1) \prod_{i=2}^{n} q(x_i|x_{1:i-1}) \sum_{x_{n+1}} q(x_{n+1}|x_{1:n}) g_\mu(x_{n+1}).$$

We show the result by induction. Consider a state node $i+1$ and assume that the constrained program (6)–(7) is equivalent to

$$\min_{\text{s.t.}(7)} \sum_{x_{1:i+1}\sim\tilde{x}_O} q(x_1) \prod_{j=2}^{i+1} q(x_j|x_{1:j-1}) h_{i+1}(x_{i+1})\,,$$

for some function h_{i+1} on $\mathcal{X}_{i+1}$, and let $q^*(X_{i+1}|X_{1:i}) := \{q^*(X_{i+1}|x_{1:i}) : x_{1:i}\}$ be the solutions to the linear optimizations

$$h_i(x_{i-1}) := \min_{q\in Q(X_{i+1}|x_{i-1})} \sum_{x_{i+1}} q(x_{i+1}|x_{1:i}) h_{i+1}(x_{i+1})$$

for different values of $X_{1:i}$. Then $q^*(X_{i+1}|X_{1:i})$ satisfies (7) and minimizes (6) w.r.t. $q(X_{i+1}|X_{1:i})$, thus

$$\min_{\text{s.t.}(7)} \sum_{x_{1:i+1}\sim\tilde{x}_O} q(x_1) \prod_{j=2}^{i+1} q(x_j|x_{1:j-1}) h_{i+1}(x_{i+1}) =$$
$$\min_{\text{s.t.}(7)} \sum_{x_{1:i}\sim\tilde{x}_O} q(x_1) \prod_{j=2}^{i} q(x_j|x_{1:j-1}) h_i(x_{i-1})\,.$$

Similarly, consider a manifest node i and assume that (6)–(7) is equivalent to

$$\min_{\text{s.t.}(7)} \sum_{x_{1:i}\sim\tilde{x}_O} q(x_1) \prod_{j=2}^{i} q(x_j|x_{1:j-1}) h_i(x_{i-1})\,. \tag{8}$$

Let $q^*(X_i|X_{1:i-1})$ be the solutions to the linear optimizations

$$h_{i-1}(x_{i-1}) := \min_{q\in Q(X_i|x_{i-1})} \sum_{x_i\sim\tilde{x}_O} q(x_i|x_{1:i-1}) h_i(x_{i-1})$$

for different values of $X_{1:i-1}$. Then, $q^*(X_i|X_{1:i-1})$ satisfies (7) and minimizes (6), therefore (8) equals

$$\min_{\text{s.t.}(7)} \sum_{x_{1:i-1}\sim\tilde{x}_O} q(x_1) \prod_{j=2}^{i-1} q(x_j|x_{1:j-1}) h_{i-1}(x_{i-1})\,.$$

The basis for $i = n$ follows trivially by setting $h_{n+1}(x_{n+1}) = g_\mu(x_{n+1})$. Thus, the unconstrained minimization in (6) (without the constraints (7)) achieves the same value of the constrained program. Moreover, it can be shown that the unconstrained program is the epistemic extension of the network (Benavoli et al., 2011), so that the result follows. □

Corollary 1. *Consider again the HMM of Theorem 1 and the same query setting, and assume that the local credal sets associated to manifest nodes i are singletons* $Q(X_i|x_{i-1}) = \{q(x_i|x_{i-1})\}$ *such that* $q(x_i|x_{i-1}) = 1$ *whenever* $x_i = x_{i-1}$. *Then the posterior lower expectations of any function f given* $\tilde{x}_O$ *is the same whether we assume epistemic irrelevance or strong independence.*

The proof of Corollary 1 follows directly from Theorem 1. Observe that since $q(x_i|x_{i-1})$ for a manifest node i are 0-1 probabilities, the HMM reduces to a(n imprecise) Markov Chain. Thus, strong and epistemic extensions coincide also in Markov Chains when evidence is before (w.r.t. the topological order) the query node. In the case of evidence after the query, de Cooman et al. (2010) have shown by a counterexample that inferences in Markov Chains are sensitive to the irrelevance concept adopted.

3.2 CREDAL TREES

Imprecise HMMs are particular cases of credal trees. De Cooman et al. (2010) showed that GBR inferences can be computed in polynomial time in credal trees under epistemic irrelevance. In this section we show that the same type of inference under strong independence is an NP-hard task.

In the intermediate steps of reductions used to show hardness results we make use of networks whose numerical parameters are specified by *(polynomial-time) computable* numbers, which might not be encodable trivially as rationals. A number r is computable if there exists a machine M_r that, for input b, runs in at most time poly(b) (the notation poly(b) denotes an arbitrary polynomial function of b) and outputs a rational number t such that $|r - t| < 2^{-b}$. Of special relevance are numbers of the form $2^{t_1}/(1 + 2^{t_2})$, with $|t_1|, |t_2|$ being rationals no greater than two, for which we can build a machine that outputs a rational t with the necessary precision in time poly(b) as follows: compute the Taylor expansions of 2^{t_1} and 2^{t_2} around zero with sufficiently many terms (depending on the value of b), and then compute the fractional expression. The following lemma ensures that any network specified with computable numbers can be approximated arbitrarily well by a network specified with rational numbers.

Lemma 1. *Consider a credal network* $\mathcal{N}$ *over n variables whose numerical parameters* $q(x_i|x_{\text{pa}(i)})$ *are specified with computable numbers encoded by their respective machines, and let b be the size of the encoding of the network. Given any rational number* $\varepsilon \geq 2^{-\text{poly}(b)}$, *we can construct in time* poly(b) *a credal network* $\mathcal{N}'$ *over the same variables whose numerical param-*

eters are all rational numbers, and such that there is a polynomial-time computable bijection (p, p') that associates any extreme p of the strong extension $\mathcal{N}$ with an extreme p' of the strong extension of $\mathcal{N}'$ satisfying $\max_{x_I \in \mathcal{X}_I} |p'(x_I) - p(x_I)| \leq \varepsilon$ for every subset of variables X_I.

Proof. Take $\mathcal{N}'$ to be equal to $\mathcal{N}$ except that each computable number r, given by its machine M_r, used in the specification of $\mathcal{N}$ is replaced by a rational t such that $|t - r| < 2^{-(n+1)(v+1)}\varepsilon$, where $v := \max_{i \in N} |\mathcal{X}_i|$ is the maximum number of values any variable can assume. Because $\varepsilon \geq 2^{-\text{poly}(b)}$, we can use M_r with input $\text{poly}(b) + (n+1)(v+1)$ to obtain t in time $O(\text{poly}(\text{poly}(b) + (n+1)(v+1))) = O(\text{poly}(b))$. Exactly one of the probability values in each pmf in $\mathcal{N}'$ is computed as one minus the sum of the other numbers to ensure that the total mass of the pmf is exactly one; its error is at most $(v-1) \cdot 2^{-(n+1)(v+1)}\varepsilon < 2^{-n(v+1)}\varepsilon$.

Let $q(x_i|x_{\text{pa}(i)})$ and $q'(x_i|x_{\text{pa}(i)})$ denote the parameters of $\mathcal{N}$ and $\mathcal{N}'$, respectively, and consider an assignment x to all variables X in $\mathcal{N}$ (or in $\mathcal{N}'$). By design $|q'(x_i|x_{\text{pa}(i)}) - q(x_i|x_{\text{pa}(i)})| \leq 2^{-n(v+1)}\varepsilon$. It follows from the binomial expansion of the factorization of $p'(x)$ that (there is a term for $p(x)$ in the expansion and $2^n - 1$ terms that an be written as a product of $2^{-n(v+1)}\varepsilon$ by numbers less than or equal to one)

$$\begin{aligned}
p'(x) &= \prod_{i \in N} q'(x_i|x_{\text{pa}(i)}) \\
&\leq \prod_{i \in N} [2^{-n(v+1)}\varepsilon + q(x_i|x_{\text{pa}(i)})] \\
&= \sum_{S \subseteq N} \prod_{i \in S} q(x_i|x_{\text{pa}(i)})[2^{-n-vn}\varepsilon]^{n-|S|} \\
&\leq 2^n 2^{-n-vn}\varepsilon + \prod_{i \in N} q(x_i|x_{\text{pa}(i)}) \\
&= p(x) + 2^{-nv}\varepsilon\,.
\end{aligned}$$

Similarly, we can show that

$$p'(x) \geq \prod_{i=1}^{n} [q(x_i|x_{\text{pa}(i)}) - 2^{-n(v+1)}\varepsilon] \geq p(x) - 2^{-nv}\varepsilon\,.$$

Thus, $\max_x |p'(x) - p(x)| \leq 2^{-nv}\varepsilon$. Now consider a subset of the variables X_I and a value $\tilde{x}_I \in \mathcal{X}_I$. Since $p'(\tilde{x}_I) = \sum_{x \sim \tilde{x}_I} p'(x)$, each term $p'(x)$ in the sum satisfies $p'(x) \leq p(x) + 2^{-nv}\varepsilon$, and there are less than $v^n \leq 2^{vn}$ terms being summed, we have that

$$p'(\tilde{x}_I) \leq \sum_{x \sim \tilde{x}_I} [p(x) + 2^{-vn}\varepsilon] \leq p(\tilde{x}_I) + \varepsilon\,.$$

An analogous argument can be used to show that $p'(\tilde{x}_I) \geq p(\tilde{x}_I) - \varepsilon$. Thus, $\max_{x_I} |p'(x_I) - p(x_I)| \leq \varepsilon$. □

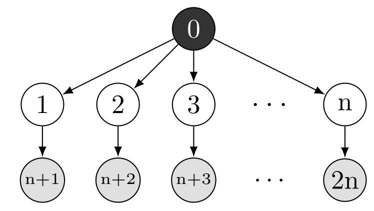


Figure 4: Credal Tree Used To Prove Theorem 2

Before proving our hardness results, we state and discuss some facts about the *partition problem*, which will be used later. The partition problem is stated as follows: given positive integers $z_1, \ldots, z_n$, decide whether there is $S \subseteq N := \{1, \ldots, n\}$ such that $\sum_{i \in S} z_i = \sum_{i \notin S} z_i$, where the notation $i \notin S$ denotes that $i \in N \setminus S$. This is a well-known NP-hard problem (Garey and Johnson, 1979). We define $v_i := z_i/z$, $i = 1, \ldots, n$, where $z := \sum_i z_i/2$, and work w.l.o.g. with the partition problem using v_i instead of z_i. Let $v_S := \sum_{i \in S} v_i$. Then, it follows for any S that $v_S = 2 - \sum_{i \notin S} v_i$. Also, if an instance of the partition problem is a *yes-instance*, there is S for which $v_S = 1$, whereas if it is a *no-instance*, then for any S, $|v_S - 1| \geq 1/(2z)$. Consider the function

$$h(v_S) = \frac{2^{-(v_S-1)} + 2^{v_S-1}}{2}\,. \tag{9}$$

Seen as a function of a continuous variable $v_S \in [0, 2]$, the function above is strictly convex, symmetric around one, and achieves the minimum value of one at $v_S = 1$. Thus, if the partition problem is a yes-instance, then $\min_S h(v_S) = 1$, while if it is a no-instance, then $\min_S h(v_S) \geq 2^{-1/(2z)-1} + 2^{1/(2z)-1} \geq 2^{(2z)^{-4}} > 1 + (2z)^{-4}/2 = 1 + 1/(32z^4)$, where the second inequality is due to Lemma 24 in (Mauá et al., 2012), and the strict inequality follows from the first-order Taylor expansion of $2^{(2z)^{-4}}$.

The following result shows that inferences under strong independence are hard even in credal trees.

Theorem 2. *Computing the GBR in credal trees under strong extension is NP-hard, even if all numerical parameters are rational numbers, and all variables are at most ternary.*

Proof. We use a reduction from the partition problem as previously described. We build a credal tree over variables $X_0, \ldots, X_{2n}$ with graph as in Figure 4. The root node is associated to the ternary variable X_0, with $\mathcal{X}_0 := \{1, 2, 3\}$ and uniform pmf $q(x_0) = 1/3$. The remaining variables are all Boolean. For $i = 1, \ldots, n$, specify the local sets $Q(X_i|x_0)$ as singletons $\{q(X_i|x_0)\}$ such that

$$q(x_i{=}1|x_0) = \begin{cases} 2^{-v_i}/(1+2^{-v_i}), & \text{if } x_0 = 1, \\ 1/(1+2^{-v_i}), & \text{if } x_0 = 2, \\ 1/2, & \text{if } x_0 = 3. \end{cases}$$

Finally, for $i = 1+n, \ldots, 2n$ specify the local credal sets such that $q(X_i|x_{i-n}) \in Q(X_i|x_{i-n})$ satisfies $q(x_i = 1|x_{i-n}) \in [\epsilon, 1]$ for all x_{i-n}, where $\epsilon = 2^{-n-3}/(64z^4)$. Consider the computation of the GBR with observed nodes $O = \{1+n, \ldots, 2n\}$, observation $\tilde{x}_O = (1, \ldots, 1) \in \mathcal{X}_O$, query node $q = 0$, and query $f(x_0) = -I(x_0{=}3)$. Using the results from (Antonucci and Zaffalon, 2006) about the *conservative inference rule*, one can show that f is minimized at an extrema $p(X)$ such that $p(x_i{=}1|x_{i-n}{=}1) \neq p(x_i{=}1|x_{i-n}{=}0)$, that is, if $p(x_i{=}1|x_{i-n}{=}1)$ is chosen to be equal to ϵ, then $p(x_i{=}1|x_{i-n}{=}0) = 1$, and vice-versa. Hence,

$$\begin{aligned} \mu = \min_p \mathbb{E}_p[f] &= -\max_p p(x_0{=}3|\tilde{x}_O) \\ &= -\max_p \left(\frac{1+\epsilon}{2}\right)^n \frac{1/3}{p(\tilde{x}_O)} \\ &= -\max_S \frac{1}{g(a_S)}\,, \end{aligned}$$

where $g(a) = 1 + (1+a)\left(\frac{2}{1+\epsilon}\right)^n \prod_{i=1}^n 1/(1+2^{-v_i})$ is defined for any real number a, and $a_S := b_S + b_{N\setminus S} - 1$ and $b_S := \prod_{i\in S}(2^{-v_i}+\epsilon)\prod_{i\notin S}(1+2^{-v_i}\epsilon)$ are defined for all $S \subseteq N$. Note that $g(a_S) > 1 + (1+a_S)2^{-n}$. It follows from the Binomial Theorem that

$$\begin{aligned} 2^{-v_S} \leq b_S &\leq (2^{-v_S} + 2^n\epsilon)(1+\epsilon)^n \\ &\leq (2^{-v_S} + 2^n\epsilon)(1+2n\epsilon) \\ &\leq 2^{-v_S} + 2^{n+2}\epsilon \end{aligned}$$

where we use the inequality $(1+r/k)^k \leq 1+2r$ valid for $r \in [0,1]$ and positive integer k (Mauá et al., 2011, Lemma 37). Thus,

$$h(v_S) - 1 \leq a_S \leq h(v_S) + 2^{n+3}\epsilon - 1\,.$$

Now if the partition problem is a yes-instance, then $a_S \leq 1/(64z^4)$, while if it is a no-instance, we have that $a_S > 1/(32z^4)$. Hence, there is a gap of at least $1/(64z^4)$ in the value of a_S between yes- and no-instances, and we can decide the partition problem by verifying whether $\mu \leq -1/g(\alpha)$, where $\alpha := 3/(128z^4)$. This proof shall be completed with the guarantee that we can approximate in polynomial time the irrational numbers used to specify the credal tree and $g(a)$ well enough so that $-1/g(\alpha)$ falls in the gap between the values of μ for yes- and no-instances. First, note that

$$g\left(\frac{1}{32z^4}\right) - g\left(\frac{1}{64z^4}\right) = \frac{1}{64z^4}\left(\frac{2}{1+\epsilon}\right)^n \prod_{i=1}^n \frac{1}{1+2^{-v_i}}\,,$$

which is greater than $2^{-n}/(64z^4)$. The gap in the value of μ is at least

$$\begin{aligned} \frac{1}{g(1/(64z^4))} - \frac{1}{g(1/(32z^4))} &= \frac{g(\frac{1}{32z^4}) - g(\frac{1}{64z^4})}{g(\frac{1}{64z^4})g(\frac{1}{32z^4})} \\ &> \frac{g(\frac{1}{32z^4}) - g(\frac{1}{64z^4})}{g(\frac{1}{32z^4})^2} \\ &> \frac{2^{-n}/(64z^4)}{(1+(1+\frac{1}{32z^4})2^{-n})^2} \\ &> \frac{2^{-n}}{4\cdot 64z^4}\,. \end{aligned}$$

So we apply Lemma 1 with $\varepsilon = \frac{1}{2}\frac{2^{-n}}{4\cdot 64z^4}$ and use the same rational numbers $q(x_i{=}1|x_o{=}2)$ as in the specification of the new network instead of the irrational values $1/(1+2^{-v_i})$ to approximate $g(\alpha)$, which guarantees that the gap will continue to exist. Alternatively, we can use the same argument as in Theorem 3 of (de Campos, 2011) to constructively find a suitable encoding for the numerical parameters and $g(\alpha)$. ☐

3.3 POLYTREES AND BEYOND

In general networks, it is still unclear which type of inferences depend on the irrelevance concept used. There is however one situation where we can show they coincide, and this is particularly important for the hardness results that we prove later on.

Lemma 2. *Consider a credal network of arbitrary topology, where all nodes are associated to precise pmfs apart from the root nodes, which are associated to vacuous credal sets. Then the result of the GBR for an arbitrary function f of a variable X_q associated to a non-root node q and no evidence is the same whether we assume epistemic irrelevance or strong independence.*

Proof. Let X_R be the variables associated root nodes (hence to vacuous local credal sets), and X_I denote the remaining variables (which are associated to singleton local credal sets). The result of the GBR under epistemic irrelevance is given by

$$\begin{aligned} \mu = \min_{p(X)} \mathbb{E}_p[f] &= \min_{p(X)} \sum_x p(x_I|x_R)p(x_R)f(x_q) \\ &= \min_{p(X)} \sum_{x_R} p(x_R) \sum_{x_I} p(x_I|x_R)f(x_q) \\ &= \min_{p(X_R)} \sum_{x_R} p(x_R)g(x_R)\,, \end{aligned}$$

where $g(x_R) := \sum_{x_I}\prod_{i\in I} q(x_i|x_{\text{pa}(i)})f(x_q)$, and $q(x_i|x_{\text{pa}(i)})$ are the single pmfs in the local credal sets of the non-root variables X_I. According to the last equality, μ is a convex combination of $g(x_R)$. Hence,

$$\mu \geq \min_{x_R} g(x_R) = \min_{x_R} \sum_{x_I}\prod_{i\in I} q(x_i|x_{\text{pa}(i)})f(x_q)\,.$$

The rightmost minimization is exactly the value of the GBR under strong independence, and since the strong extension is contained in the epistemic extension, the inequality above is tight. □

The above result will be used in combination with hardness results for strong credal networks to demonstrate that computing the GBR under epistemic irrelevance is also hard. First, we focus on singly connected networks.

For polytrees, the next theorem shows that inferences in credal networks, either under epistemic irrelevance or strong independence, are NP-hard, even if variables are at most ternary. If we allowed variables to have any arbitrary finite number of states, then this result would follow from the proof of NP-hardness of inferences in polytrees given by de Campos and Cozman (2005), because the polytree presented there is similar to the one in Figure 5 with no evidence and query in the last (topological) node. By Lemma 2, we could use an inference in the epistemic polytree to solve the same inference, demonstrating that such inference is NP-hard too. In this paper we devise a stronger result, as the hardness is shown even when variables are at most ternary. For this purpose, we perform a polynomial-time reduction from the partition problem. A much similar reduction has been used to show that selecting optimal strategies in limited memory influence diagrams is NP-hard (Mauá et al., 2012). While the reduction used here closely resembles the reduction used in that work, due to a technicality we cannot directly use that result (mainly because the influence diagram could have multiple utility nodes, which would much complicate the reduction). Instead, we directly reduce an instance of the partition problem to a computation of the GBR without evidence in a credal polytree whose non-root nodes are all associated to precise pmfs and root nodes are associated to vacuous credal sets. By using this reduction in conjunction with Lemma 2, we prove the hardness of GBR computations also under epistemic irrelevance.

Theorem 3. *Given a credal polytree, computing the GBR with a function f of the query variable X_q and no evidence is NP-hard, whether we assume epistemic irrelevance or strong independence, even if all variables are (at most) ternary and all numbers are rational.*

Proof. We build a credal polytree with underlying graph as in Figure 5. The variables (associated to nodes) on the upper row are Boolean and vacuous, namely $X_1, \dots, X_n$, while the remaining variables $X_{n+1}, \dots, X_{2n+1}$ are ternary and associated to singleton local credal sets such that $Q(X_{n+1})$ contains a uniform pmf $q(x_{n+1}) = 1/3$, and, for $i = n+2, \dots, 2n+1$, $Q(X_i|x_{i-1}, x_{i-n-1})$

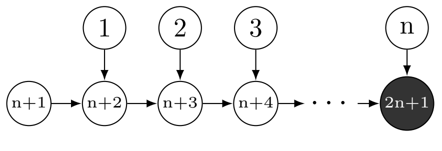


Figure 5: Polytree Used To Prove Theorem 3

Table 2: Local Pmfs Used To Prove Theorem 3

$q(x_i \vert x_{i-1}, x_{i-n-1})$	$x_i=1$	$x_i=2$	$x_i=3$
$x_{i-1}=1, x_{i-n-1}=1$	2^{-v_i}	0	$1-2^{-v_i}$
$x_{i-1}=2, x_{i-n-1}=1$	0	1	0
$x_{i-1}=3, x_{i-n-1}=1$	0	0	1
$x_{i-1}=1, x_{i-n-1}=0$	1	0	0
$x_{i-1}=2, x_{i-n-1}=0$	0	2^{-v_i}	$1-2^{-v_i}$
$x_{i-1}=3, x_{i-n-1}=0$	0	0	1

contains the pmf $q(X_i|x_{i-1}, x_{i-n-1})$ as specified in Table 2. Consider a joint pmf $p(X)$ which is an extreme of the strong extension of the network. One can show that $p(x) = q(x_{n+1}) \prod_{i=n+2}^{2n+1} q(x_i|x_{i-1}, x_{i-n-1}) \prod_{i \in S} I(x_i = 1) \prod_{i \notin S} I(x_i = 0)$ for all x, where $S \subseteq N$. Thus, $p(x)$ is equal to $\frac{1}{3} \prod_{i=n+2}^{2n+1} q(x_i|x_{i-1}, x_{i-n-1})$ if $x_S = 1$ and $x_{N \setminus S} = 0$, and otherwise vanishes. It follows that $p(x_{2n+1} = 1) = \sum_x p(x) = (2^{-v_S})/3$ and $p(x_{2n+1} = 2) = (2^{v_S - 2})/3$. Let $\alpha := (1 + z^{-4}/64)/3$. By computing the GBR with query $f(x_{2n+1}) = I(x_{2n+1} = 1) + I(x_{2n+1} = 2)$ and no evidence, we can decide the partition problem, as $\min_p \mathbb{E}[f] = \min_S h(v_S)/3 \leq \alpha$, if and only if the partition problem is a yes-instance. According to Lemma 2, this result does not change if we assume epistemic irrelevance. It remains to show that we can polynomially encode the numbers 2^{-v_i}. This is done by applying Lemma 1 with a small enough ε computable in time polynomial in the size of the partition problem: $\varepsilon = 1/(3 \cdot 64 z^4)$ suffices. □

The hardness of inference in multiply connected credal networks under epistemic irrelevance comes from the fact that general credal networks under strong independence can always be efficiently mapped to credal networks under strong independence with all non-root nodes precise and vacuous root nodes. Because such inferences in general credal networks under strong independence are NP$^{\text{PP}}$-hard (Cozman et al., 2004), and because Lemma 2 demonstrates that marginal inferences without evidence in these networks are equivalent to the same inference in credal networks under epistemic extension, the hardness result is obtained also for epistemic networks. For completeness, we write the complete proof of such result using a reduction from the E-MAJSAT problem, since previous work has only provided a sketch of such proof (Cozman et al., 2004). The proof differs only slightly from the proof of $\mathrm{NP}^{\mathrm{PP}}$-hardness of MAP inference in Bayesian networks given by Park and Darwiche (2004).

Theorem 4. *Computing the GBR in credal networks under either epistemic irrelevance or strong independence is NP^{PP}-hard even if all variables are binary.*

Proof. The hardness result follows from a reduction from E-MAJSAT. Given a propositional formula ϕ over Boolean variables $Z_1, \ldots, Z_n$ and an integer $1 \leq k < n$, the E-MAJSAT is the problem of deciding whether there exists an assignment to $Z_1, \ldots, Z_k$ such that the majority of the assignments to the remaining variables $Z_{k+1}, \ldots, Z_n$ satisfy ϕ. The reduction proceed as follows. Create a credal network over Boolean variables $X = (X_1, \ldots, X_k, X_{k+1}, \ldots, X_n)$ such that $X_1, \ldots, X_k$ are associated to root nodes and vacuous credal sets, and $X_{k+1}, \ldots, X_n$ are associated to non-root nodes and have uniform pmfs. The root variables act as selectors for the Boolean variables in the propositional formula. Each variable X_i is associated to the Boolean variable Z_i of the original formula ϕ. Build one new binary variable X_i (using a suitable sequence of numbers $i = n+1, n+2, \ldots$) for each operator in the Boolean formula ϕ of the E-MAJSAT problem such that X_i has as parents its operands, that is, for logical operations $(X_a \wedge X_b)$ and $(X_a \vee X_b)$, with $a, b < i$, X_i has as parents the two operands X_a and X_b and is associated to a singleton credal set containing the pmf $q(x_i|x_a, x_b) = I(x_i = x_a \circ x_b)$, where $\circ$ denotes the respective binary operation; for the operation $(\neg x_a)$, with $a < i$, X_i has a single parent X_a and is associated to a singleton local credal set containing the pmf $q(x_i|x_a) = I(x_i \neq x_a)$. There is more than one way to build such a network, depending on the order one evaluates the operations in the Boolean formula, and any valid evaluation order can be used. The final network encodes a circuit for evaluating the formula ϕ.

Let t be the last node in the network in topological order; variable X_t represents the satisfiability of the whole formula. Consider a joint pmf $p(x) = \prod_{i \in N} q(x_i|x_{\mathrm{pa}(i)})$, for some choice of pmfs from the extrema of local credal sets. By design, there is a single $x_{1:k} \in \mathcal{X}_{1:k}$ such that $\prod_{i=1}^{k} q(x_i)$ evaluates to one. Let $\tilde{x}_{1:k}$ be such (joint) value. We have that

$$\begin{aligned} p(X_t = 1) &= \sum_{x \sim x_t} p(x_t = 1|x_I) p(x_I|x_R) \prod_{i=1}^{n} q(x_i) \\ &= \frac{1}{2^{n-k}} \sum_{x \sim \tilde{x}_{1:k}} p(x_t = 1|x_I) p(x_I|x_R) , \end{aligned}$$

where X_I are the non-root variables that represent the logical operations in the formula ϕ apart from X_t, $X_R = (X_1, \ldots, X_n)$ are the root variables associated to Boolean variables in ϕ. The value of $p(x_t = 1|x_I)p(x_I|x_R)$ is equal to one if and only if the assignment x_R satisfies ϕ (by construction) and is zero otherwise. Thus, $p(X_t = 1) = \#\mathrm{SAT}/2^{n-k}$, where #SAT is the number of assignments to the variables $Z_{k+1}, \ldots, Z_n$ that satisfy ϕ with the first k Boolean variables set to $\tilde{x}_{1:k}$. By maximizing over the possible choices of degenerate pmfs $q(X_i)$, $i = 1, \ldots, k$, we have that $\min_p p(x_t = 0) < 1/2$ if and only if there is an assignment to the first k variables $Z_1, \ldots, Z_k$ such that more than half of the assignments to the remaining $n - k$ variables $Z_{k+1}, \ldots, Z_n$ satisfy the formula ϕ. $\square$

4 CONCLUSION

In this paper, we have showed new computational complexity results for inference in credal networks under both epistemic irrelevance and strong independence. There are three main contributions. First, by exploiting the relations between these two irrelevance concepts in HMM-like credal networks, we have shown that predictive inferences under strong independence can be computed in polynomial time using the same algorithm developed for epistemic credal trees. To complement such result, we have proved that inferences with strong independence in general trees are NP-hard even if all variables are (at most) ternary, which shows that it is unlikely that more general polynomial-time algorithms for inferences under strong independence will ever exist. Moreover, using the relation between strong and epistemic irrelevance concepts in networks where the imprecision appears only in root nodes (defined by vacuous credal sets), we were able to prove that inferences in polytrees under epistemic irrelevance are NP-hard, even if all variables are (at most) ternary. This result closes the gap between known polynomial-time algorithms (which were known for trees and some polytrees) and potentially any more complicated network. To the best of our knowledge, these complexity results were all open, specially for the case of epistemic irrelevance.

Acknowledgements

This work has been partly supported by the Swiss NSF grants no. 200020_137680/1 and 200020_134759/1, and the Hasler Foundation grant no. 10030.

References

Antonucci, A., Brühlmann, R., Piatti, A., and Zaffalon, M. (2009). Credal networks for military identification problems. *International Journal of Approximate Reasoning*, 50(4):666–679.

Antonucci, A. and Piatti, A. (2009). Modeling unreliable observations in Bayesian networks by credal networks. In *Proceedings of the 3rd International Conference on Scalable Uncertainty Management (SUM)*, volume 5785 of *Lecture Notes in Computer Science*, pages 28–39. Springer.

Antonucci, A., Piatti, A., and Zaffalon, M. (2007). Credal networks for operational risk measurement and management. In *International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES)*, volume LNCS 4693, pages 604–611.

Antonucci, A. and Zaffalon, M. (2006). Equivalence between Bayesian and credal nets on an updating problem. In *Proceedings of 3rd International Conference on Soft Methods in Probability and Statistics (SMPS)*, pages 223–230. Springer.

Benavoli, A., Zaffalon, M., and Miranda, E. (2011). Robust filtering through coherent lower previsions. *IEEE Transactions on Automatic Control*, 56(7):1567–1581.

Cozman, F. G. (2000). Credal networks. *Artificial Intelligence*, 120(2):199–233.

Cozman, F. G., De Campos, C. P., Ide, J. S., and da Rocha, J. C. F. (2004). Propositional and relational Bayesian networks associated with imprecise and qualitative probabilistic assessments. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 104–111. AUAI Press.

de Campos, C. P. (2011). New complexity results for MAP in Bayesian networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2100–2106. AAAI Press.

de Campos, C. P. and Cozman, F. G. (2005). The inferential complexity of Bayesian and credal networks. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1313–1318, San Francisco, CA, USA. Morgan Kaufmann.

de Cooman, G., Hermans, F., Antonucci, A., and Zaffalon, M. (2010). Epistemic irrelevance in credal nets: the case of imprecise Markov trees. *International Journal of Approximate Reasoning*, 51(9):1029–1052.

de Cooman, G. and Troffaes, M. (2004). Coherent lower previsions in systems modelling : products and aggregation rules. *Reliability engineering & system safety*, 85(1-3):113–134.

Fagiuoli, E. and Zaffalon, M. (1998). 2U: An exact interval propagation algorithm for polytrees with binary variables. *Artificial Intelligence*, 106(1):77–107.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman.

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models.* MIT press.

Kwisthout, J. and van der Gaag, L. C. (2008). The computational complexity of sensitivity analysis and parameter tuning. In *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 349–356. AUAI Press.

Levi, I. (1980). *The Enterprise of Knowledge.* MIT Press, London.

Mauá, D. D., de Campos, C. P., and Zaffalon, M. (2011). Solving limited memory influence diagrams. *CoRR*, abs/1109.1754.

Mauá, D. D., de Campos, C. P., and Zaffalon, M. (2012). Solving limited memory influence diagrams. *Journal of Artificial Intelligence Research*, 44:97–140.

Mauá, D. D., de Campos, C. P., and Zaffalon, M. (2012). Updating credal networks is approximable in polynomial time. *International Journal of Approximate Reasoning*, 53(8):1183–1199.

Park, J. D. and Darwiche, A. (2004). Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research*, 21:101–133.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, California.

Piatti, A., Antonucci, A., and Zaffalon, M. (2010). *Building Knowledge-Based Systems by Credal Networks: A Tutorial.* Nova Science.

Walley, P. (1991). *Statistical Reasoning with Imprecise Probabilities.* Chapman and Hall, London.

Zaffalon, M. and Miranda, E. (2009). Conservative inference rule for uncertain reasoning under incompleteness. *Journal of Artificial Intelligence Research*, 34:757–821.

Learning Periodic Human Behaviour Models from Sparse Data for Crowdsourcing Aid Delivery in Developing Countries

James McInerney, Alex Rogers, Nicholas R. Jennings
University of Southampton, Southampton, SO17 1BJ, UK
{jem1c10,acr,nrj}@ecs.soton.ac.uk

Abstract

In many developing countries, half the population lives in rural locations, where access to essentials such as school materials, mosquito nets, and medical supplies is restricted. We propose an alternative method of distribution (to standard road delivery) in which the existing mobility habits of a local population are leveraged to deliver aid, which raises two technical challenges in the areas optimisation and learning. For optimisation, a standard Markov decision process applied to this problem is intractable, so we provide an exact formulation that takes advantage of the periodicities in human location behaviour. To learn such behaviour models from sparse data (i.e., cell tower observations), we develop a Bayesian model of human mobility. Using real cell tower data of the mobility behaviour of 50,000 individuals in Ivory Coast, we find that our model outperforms the state of the art approaches in mobility prediction by at least 25% (in held-out data likelihood). Furthermore, when incorporating mobility prediction with our MDP approach, we find a 81.3% reduction in total delivery time versus routine planning that minimises just the number of participants in the solution path.

1 INTRODUCTION

In many developing countries (e.g., Ivory Coast, Ghana, Liberia, Nigeria), half the population lives in rural locations [5], where accessibility to school materials, medical supplies, mosquito nets, and clothing is restricted. Distribution to these locations typically requires direct road transport, which is time consuming and requires bulk volume to be cost effective. In response to these limitations, distributed methods of aid distribution have emerged in recent years.

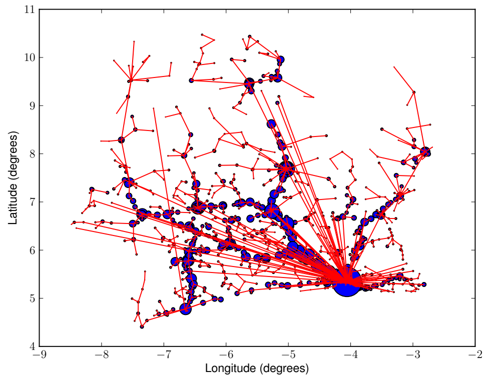


Figure 1: Minimum spanning tree between cell towers in Ivory Coast, where connections are defined by common visitors, and the size of node represents its betweenness centrality (i.e., the number of times the location appears in the shortest path for all possible delivery paths).

For example, Pack For a Purpose[1] is a non-profit organisation that asks tourists who already have a trip planned for one of 47 developing countries to bring small items (e.g., pencils, deflated soccer balls, stethoscopes) in their spare luggage capacity. Another scheme is Pelican Post[2], which asks donors to send books by post to developing countries. These are promising schemes. However, they fail during periods of conflict, (e.g., post-electoral violence in Ivory Coast in 2011) and are reliant on direct outsider support, when it is arguably preferable to empower local populations wherever possible.

In this work, we propose a new distribution method that uses the natural mobility of a local population to distribute physical packages from one location to another. In more detail, we wish to take advantage of the pre-existing mobility routines of a set of local participants by asking them to pick up a package from one exchange point (at a location that they normally visit, at a time that they normally

[1]`http://www.packforapurpose.org`
[2]`http://www.pelican-post.org`

visit it) and then drop it off at another exchange point (e.g., a lockbox or affiliate store) that is also part of their regular mobility. By chaining together the mobility of several participants, we may cover a large area, possibly a whole country, without having to deploy more expensive and time consuming infrastructure.

While potentially appealing, this vision of crowdsourcing physical package delivery faces two signiÞcant technical barriers in optimisation and learning[3].

In optimisation, the possible delay between stages in the package's journey is unbounded, since the delay introduced by each participant is unknown and has no upper limit. This makes it infeasible to optimise the selection of participants and the package route (given a specific delivery problem specifying the start time, source location, and destination location), as delays propagate through the system [11]. In general, routing under delay uncertainty is a #P-hard problem to solve optimally [17]. Therefore, we formulate the decision problem in a way that takes advantage of the periodicities in human location behaviour to derive an exact solution.

In learning, the historical movements of individuals may be obtained from cell tower connections registered by mobile devices, which have widespread adoption across the developing world. However, such mobility data is sparse: it is limited in duration (i.e., we may only have a few week's worth of data from each participant) and, crucially, cell tower readings are taken only when a call or text message is exchanged from the phone, so there are large periods when no location of an individual is registered at all. Yet, existing methods for mobility prediction rely on large quantities (covering several weeks) of fairly continuous stream of location readings (either from GPS or constant cell tower monitoring) [23, 6, 14]. To overcome this, we develop a robust Bayesian model of individual mobility that can be learnt from cell tower records spanning only short periods of time with sporadic observability.

In more detail, we make the following three contributions:

- We advance the state of the art in route planning in delay networks by developing an approach that works well with the uncertainties caused by routine human behaviour. Specifically, we show that an exact and tractable solution is possible when using a mobility model belonging to the broad class of *temporal periodic* prediction models. Under this assumption, we show that we can formulate the problem as a Markov decision process (MDP) in which the number of states grows linearly in the number of locations, making the overall algorithm polynomial when using a standard MDP solving method (e.g., linear programming, policy iteration) [20]. Using our approach, simulations indicate that source-to-destination delivery time is reduced by an average of 81.3% compared to choosing the shortest path (which naïvely minimises the number of intermediate stages in the package's journey).

- To provide accurate transition probabilities to the MDP[4], we present a Bayesian nonparametric mixture model approach to learning the mobility behaviour of individuals from very sparse observations. We show how this model can be formulated as a series of Bernoulli trials and directly incorporated into the MDP. Using real cell tower data from 50,000 people in Ivory Coast (provided by Orange), we find at least a 25% improvement in held-out data likelihood when compared to two state-of-the-art approaches for human location behaviour prediction (a variable-order Markov model with prediction by partial matching [25] and a daily periodic finite mixture model [4])

- We use the Orange dataset to show that peer-to-peer package delivery is feasible under three key criteria. In particular, we show that the size of participant pool only needs to be of the order of several thousand to get at least an 80% coverage of the country (out of a total area 320,000 km^2). Furthermore, each solution path (i.e., chain of participants to deliver a package) is between 2-4 people. Finally, these requirements are only mildly worsened when considering only rural destinations for delivery.

The rest of the paper is structured as follows. First, in Section 2, we consider previous work related to the problem of learning human mobility patterns and optimising under uncertainty of human behaviour. In Section 3, we present our approach, starting with how we make optimal decisions with respect to the choice of participants and locations for any given delivery problem in Section 3.1. Then, in Section 3.2, we present a learning model that deals with sparse observations. In Section 4, we evaluate the feasibility of the scenario before evaluating our approach to learning and optimisation against several state of the art benchmarks. We draw conclusions and outline future work in Section 5.

2 RELATED WORK

The idea of distribution using the natural mobility of a group of people is a reoccurring theme in content distribution using mobile *ad-hoc* networks. For example, Keller et al. (2012) used physical bluetooth proximity data from the mobile phones of a group of people, to initiate exchanges

[3]In addition to social issues such as trust (e.g., theft or loss) that we only consider briefly, in Section 4.4.

[4]N.B., the transition probabilities in the MDP are *not* the same as the transition probabilities of the mobility of any individual participant.

of songs between individuals, but without considering prediction or multi-hop routes (i.e., going via one or more intermediaries) [10]. Cherubini et al. (2010) explored physical package peer-to-peer delivery, but only tested simple heuristics such as "transfer the package to someone who is, on average, closer to the target location than you" [3]. Vukadinovic et al. (2009) proposed a queuing model of the flow of pedestrian crowds to distribute content among mobile phones [27]. Now, all these works attempt to capture short term movements of individuals in crowds, which is a distinct and different problem to extracting *routine* mobility patterns. Specifically, in our work, there is a direct line of assumptions going from the raw historical data to decision-making about distribution (via learning and the formulation of transition probabilities in the MDP) that is not present in such work. A notable exception is by Liu and Wu (2011), who used class attendance data to model pairwise encounters between individuals for data transfer across an *ad hoc* network [13]. Like our work, they also take advantage of cyclic behaviour to find tractable routing solutions, however, their pairwise approach means that their algorithm scales $O(p^2)$ in the number (p) of participants in the network, while our approach scales polynomially only in the number of locations. In general, content distribution approaches often rely on the fact that content may be copied and can exist concurrently on multiple devices, making them less applicable for our routing problem.

Another type of diffusion that attracts intense research interest is the study of the spread of infectious diseases. Epidemiologists look at the mobility dynamics of a population to identify source regions (from which disease is spread), and likely importation regions (to which disease is spread). For example, Wesolowski et al. (2012) used one year of cell phone data of millions of people to model the human movement between different regions in Nairobi [28]. They considered a graph in which the weight of the edges represents the quantity of people travelling between different locations. Hufnagel et al. (2004) considered a global model of human movement using passenger numbers for flights between the 500 largest airports in the world [9]. Such work is concerned with *aggregate* mobility, in contrast, we are interested in *individual* mobility, because, eventually, we need to ask specific people to contribute. Furthermore, we consider a full decision-making model, in addition to a purely descriptive model of human location behaviour.

The problem of robust route planning under uncertainty resembles the *Canadian traveller* problem (also known as the *bridge* problem) [18], in which the costs of the edges in a graph are random variables that are observed only as the nodes are visited. The name originates from the concept of a traveller who has to plan a journey between two locations, where the costs of outgoing edges are random variables that are only observed as a graph is traversed. This differs from our problem because the Canadian traveller assumes that path costs are independent of one another, while we have dependencies between costs as well, i.e., the delay outcome of an earlier stage in the chain affects the delay of later stages. An additional difference is that we observe the random variables, indicating delay between locations, only *after* the package has completed each intermediate step.

Learning routine mobility models has typically been a separate problem from optimisation. Approaches range from purely temporal ([15, 23, 24]), spatial ([7, 25]), to a combination of both ([6]). Existing datasets that are widely available have tended to contain approximately continuously recorded cell towers or GPS (e.g., the Reality Mining dataset recorded the cell tower every few minutes [6]; the Nokia dataset recorded GPS every few minutes also [12]). This has inspired many methods that work well on continuous location updates, but which do not perform as well as their headline accuracy (when predicting future location behaviour) on sparse data. We address this issue in our work. Given their ability to refine the model as more data arrives, nonparametric Bayesian methods are surprisingly rare in the literature on predicting human location behaviour. Chen et al. (2012) used a Gaussian process to model congestion on road networks, while Gao et al. (2012) used a hierarchical Pitman Yor process to model check-in behaviour on location-based social networks [2, 7].

Finally, crowdsourcing teams of participants who function as a chain to achieve a single goal resembles the idea behind the winning entries to the DARPA Red Balloon Challenge [19] and the Tag Challenge [21]. This work is primarily concerned with the problem of recruiting individuals and verifying their reports, which requires designing economic mechanisms. In this work, we assume recruitment can be done beforehand by an appropriate method (i.e., we do not address it here) but we do investigate *how many* participants are required for satisfactory delivery results.

3 DECISION-MAKING WITH UNCERTAIN HUMAN LOCATIONS

In this section, we present our approaches towards optimisation and learning with uncertain human behaviour in the package delivery scenario. Specifically, in Section 3.1, we show how it is possible to tractably find an exact optimal solution to routing under delay uncertainty, given a wide class of mobility model (which we define as *temporal periodic* models). In Section 3.2, we give more detail on our probabilistic mobility model that is designed to function well with sparse mobile phone datasets, and provides the predictions used in the optimisation.

3.1 THE OPTIMISATION PROBLEM

We formulate the optimisation problem sketched in Section 1 as an MDP, as this provides a principled way of making decisions under uncertainty. Decisions in this scenario must specify which participants to ask to pick up the package, from where they should pick it up, as well as the drop-off location. We assume the delay between pick-up and drop-off is outside the planner's control (so we treat it as a random variable here), and completely up to the participant who, when asked, does this according to his/her routine schedule.

In general, an MDP is defined as a tuple (S, A, R, T) where S is a set of states, A is a set of available actions for each state, $R(s, a, s')$ is the function that specifies the cost of doing action $a \in A$ to get from state s to s', and $T(a, s, s')$ is the probability of getting from state s to s' when performing action a[5]. The solution to an MDP consists of an optimal policy, $q(s)$, that specifies the best action to perform for any given state s. Ancillary to this function is the value function, G, which gives the expected value for any state (given that the optimal action is performed). We consider each of A, S, R, and T in turn.

3.1.1 Set of Actions A

We assume that the planner has no direct control over the delay (it is up to the participant's schedule) but we assume that we are guaranteed to eventually reach location w, when performing action a (going to location w), and that the arrival time is revealed only after performing each action, resulting in a transition to state (v, t_v), with unknown arrival time t_v. Given the one-to-one mapping of actions and locations (specifying the destination location) we treat locations as synonymous with actions.

3.1.2 Set of States S

We define the set of states S in the MDP as the set of tuples describing the possible locations and times (v, t_v) (respectively) of the package. This results in the set $S = \{(v, t_v)|v \in V, t_v = 1, 2, 3, ...\}$. We assume discrete time t to capture the required detail in the scenario without the need for more complex continuous time reasoning. However, even in the discrete time case, we see that there is an unbounded number of states in S because the delay in moving between locations is unbounded. This makes the standard MDP formulation intractable.

To overcome large state spaces, there are a few general approaches such as sampling methods or value approximation (in which values are computed from features of the states) [29]. One time-specific approach is to truncate the range of values for t to find an approximation for the optimal policy [26]. However, the number of states grows as a factor of this truncation limit, so more exact approximations must be traded off with computation time.

Instead, we find an exact solution under an additional assumption about the mobility model used to produce the probabilistic delays. Specifically, we show that for a large class of mobility models, namely *periodic temporal* models, the probability of delay, $pr(t_w - t_v|v, t_v, w)$ in going from state (v, t_v) to (w, t_w) is periodic in t_v. This results in an MDP with a linear number of states in the number of locations. This assumption is suitable for optimisation in delay networks, since it is precisely the periodic temporal class of mobility model that is most useful in predicting and planning several days in advance, since short term spatial correlations (e.g., a participant tends to go home after visiting the market, or always goes to the city centre after travelling along a particular road) do not have much effect beyond several hours. However, this assumption of temporal periodicity means that we cannot incorporate the most recent observations into the model, which may provide a benefit in optimising decisions to be made in the very near future. Under this assumption, we now establish linearity in the number of locations.

Theorem 1. Let S be the set of states $\{(v, t_v)|v \in V, t_v = 1, 2, 3, ...\}$ in an MDP. If $pr(v|t_v)$ is a periodic function (defining H as the number of possible values it can take) in discrete t_v $(\forall v)$, then the number of states is linear in the number of locations, i.e., $|S| = H\,|V|$.

Proof: Let $pr(v|t_v)$ be the probability that a given participant is at location v at time t_v, obtained from a mobility model (which, we emphasise, describes individual behaviour and is distinct from the transition function T of the MDP defined in Section 3.1.3). Since t_v is discrete, we can repeat Bernoulli trials from the distribution $r_{d_v} \sim Bern(pr(v|t_v + d_v))$ for increasing $d_v = 1, 2, 3, ...$ until we get $r = 1$. This is a standard formulation (equivalent to repeated tosses of biased coins), with $pr(d_v|t_v) = pr(v|t_v + d_v) \prod_{d'_v=1}^{d_v-1} (1 - pr(v|t_v + d'_v))$. Since $pr(v|t_v)$ is periodic in t_v, with a maximum of H distinct values, the probability of delay, $pr(w|t_v + d_v)$, *from* any next location w (reachable from v) is also periodic for arbitrary delay d_v. Therefore, $pr((t_v + d_v) \bmod H)$ is a sufficient statistic for $pr(d_w|t_v + d_v)$ (the probability of delay d_w from w), clearly taking at most H values. Using the Markov property of MDPs, only H states are required for each location v (for arbitrary v), resulting in $H\,|V|$ states overall. □

Unlike a truncation parameter, we can easily set H for the specific application of the delay network that needs to be modelled, without bias (i.e., without underestimating the delay). For package delivery, we found it sufficient to set $H = 14$ per week, by considering the probability of a participant dropping off or picking up the package in slots of half a day. Therefore, the state space is now $S = \{(v, t)|v \in V, t \in [1, 14]\}$.

[5]It is typical to include a time discount factor for future rewards in an MDP, however, this assumption makes less sense when utility is a function of delay. Therefore, we omit it in our model.

3.1.3 Cost Function R and Transition Function T

The delay in going from location v to location w is the cost function $R(s, a, s')$, where $s = (v, t_v)$, $s' = (w, t_w)$, and a is the action of routing the package to w. The MDP requires a single cost for each state s and action a pair (marginalising over the destination action), yet we have many participants who can potentially perform that action (i.e., who routinely visit both v and w locations). We define the *best person* as the one who minimises $p^* = \arg\min_i \{\mathbb{E}(d_{v,w}|t_v, i) + \sum_w c_w pr(t_w|t_v, i)|p_i \in P\}$, the cost of going from location v to w plus the expected cost of c_w (the total cost at state (w, t_w)). The cost function R is then the sum of delays for the best person to pick the package up at location v, and drop the package off at w:

$$\begin{aligned}
&R((v,t_v),w,(w,t_w)) \\
&\qquad = \mathbb{E}(d_v|t_v \bmod H) + \mathbb{E}(d_w|t_v + d_v \bmod H) \\
&\qquad = \sum_{i=0}^{\infty} W^i(Hi + d_v)pr(d_v) \prod_{d'_v=1}^{d_v-1} (1 - pr(d'_v)) \\
&\qquad + \sum_{i=0}^{\infty} W_w^i(Hi + d_w)pr(d_w) \prod_{d'_w=1}^{d_w-1} (1 - pr(d'_w))
\end{aligned} \tag{1}$$

where $W_v = \prod_{d'_v=1}^{H} (1 - pr(d'_v))$ and $W_w = \prod_{d'_w=1}^{H} (1 - pr(d'_w))$, with the respective interpretations being the probability of the participant *not* visiting the start and end locations (respectively) for an entire period. We now find the geometric sum:

$$\begin{aligned}
&R((v,t_v),w,(w,t_w)) = \\
&\left(\frac{d_v}{1-W_v} + \frac{W_v H}{(1-W_v)^2}\right) pr(d_v|t_v) \prod_{d'_v=1}^{d_v-1} (1 - pr(d'_v|t_v)) \\
&+ \left(\frac{d_w}{1-W_w} + \frac{W_w H}{(1-W_w)^2}\right) pr(d_w|t_v + d_v)\cdot \\
&\cdot \prod_{d'_w=1}^{d_w-1} (1 - pr(d'_w|t_v + d_v))
\end{aligned} \tag{2}$$

The transition function $T(a, s, s')$ may be found in a similar way, but by considering only whole multiples of the given delay:

$$T(w,(v,t_v),(w,t_w)) = \sum_{d_v=1}^{H} pr(d_v|t_v)pr(d_w|t_v, d_v) \tag{3}$$

where $d = (t_w - t_v) \bmod H$, and we have marginalised out the uncertainty about d_v (the uncertainty in pick-up delay).

We next address the problem of learning mobility models for individuals, which provides the probability of presence that defined the Bernoulli trial used in Equations 2 and 3.

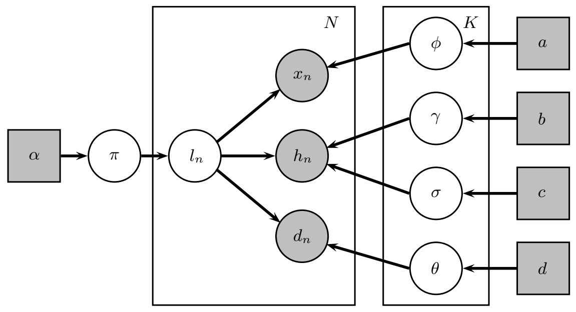


Figure 2: Graphical structure of the Dirichlet process location model, showing conditional independence between the random variables. Shaded nodes are observable and square nodes are fixed values.

3.2 MODEL FOR LEARNING HUMAN MOBILITY FROM CELL PHONE DATA

We now focus on the problem of getting an accurate predictive probability density of presence for any location given the participant and the time $pr(v|i, t_v)$, from which the probability of delay can be derived and used in optimisation (as described in Section 3.1). The Orange dataset consists of a set of tuples for each participant $p_i \in P$ of the form (i, x_i, t_i) indicating that participant i was observed near cell tower x_i (discrete) at date and time t_i (continuous). There are three main factors that influence the design of the model:

1. **Cell allocation noise**
   The cell tower observations provide discrete measurements on the individual's likely location. However, there may be a choice of several towers that the phone can connect to (especially in urban environments) at any single location. This allocation is decided by outside factors that we treat as noise (i.e., the network operator's optimal allocation of phones to towers). Our approach needs to isolate the human presence information in the cell tower allocation to phones and ignore other factors. This implies the need to infer the locations, each of which may be statistically associated with several cell towers.
2. **Sporadic observations**
   Since the cell tower is only recorded in this dataset when a phone call or text is made (about 7 times a day, on average) approaches that were designed to be used on continuously collected location data (e.g. eigenvectors [22, 6], variable-order Markov models [25], linear embedding [23]) are not likely to be effective (which we confirm in Section 4.2). We therefore need a method that can fill in (extrapolate from other observations) large periods of no observability.
3. **Short duration**
   The data on each individual covers a period of only 2

weeks. This, combined with the fact that each day may have only a few (or zero) observations, makes learning challenging. Overfitting is a danger when the training data (i.e., the 2 weeks of observations) contains characteristics that do not generalise to the rest of the individual's behaviour (i.e., beyond 2 weeks).

These considerations suggest the use of the Bayesian framework, which allows us to assume the existence of latent variables that abstract away from the variability of cell allocation (Factor 1), and make custom assumptions about the smoothness of location (Factor 2). Furthermore, Bayesian non-parametric methods can provide us with powerful guards against overfitting (Factor 3).

In more detail, we assume the existence of latent discrete locations, l_n, that are associated with each observation (x_n, t_n), and correspond to places in the individual's routine life (e.g., home, work). Mixture modelling is a well established method for inferring latent discrete variables, but the standard approach requires the specification of the number of locations [1]. Therefore, we use a Dirichlet process mixture model (a non-parametric approach) that allows us to also infer the number of locations, K [16]. This is important because setting K too high (manually) will cause the model to overfit the data.

To address the problem of filling in large periods of missing data, we assume that behaviour is periodic, as is common in other routine mobility models [22, 23]. Specifically, we assume both weekly and daily periodicities in behaviour. In the model, we achieve this by decomposing the date/time observation t_n to the discrete day of the week, d_n, and continuous hour of the day h_n. The practical implications of this choice are explored briefly in Section 4.4.

A full generative model for location observations of each individual is therefore the following:

$$\boldsymbol{\pi} \sim DP(\alpha) \tag{4}$$

for each latent location k :

$$\boldsymbol{\phi_k} \sim Dir(a),\ \gamma_k \sim \mathcal{N}(b) \tag{5}$$

$$\omega_k \sim IG(c),\ \boldsymbol{\theta_k} \sim Dir(d) \tag{6}$$

for each observation n :

$$l_n \sim \mathcal{M}(\boldsymbol{\pi}),\ x_n \sim \mathcal{M}(\boldsymbol{\phi_{l_n}}) \tag{7}$$

$$h_n \sim \mathcal{N}(\gamma_{l_n}, \omega_{l_n}),\ d_n \sim \mathcal{M}(\boldsymbol{\theta_{l_n}}) \tag{8}$$

where, first, distribution $\boldsymbol{\pi}$ over latent locations is drawn from a Dirichlet process (Equation 4) that defines the prior probability of each location in the dataset. Second, the four parameters to the model $\phi, \gamma, \omega, \theta$ are drawn from their prior distributions (Dirichlet, normal, inverse-gamma, and Dirichlet, respectively) in Equations 5-6 [1]. These priors were chosen for their conjugacy to the parameter distributions, making the model simpler to infer. Thirdly, for each observation, latent location l_n is drawn (Equation 7), and this location defines all the observable information in the dataset (x_n, the cell tower, h_n the continuous hour observation, and d_n, the day of the week). Since x_n and d_n are discrete observations, they can be drawn from multinomials, while h_n (the continuous hour of the day) is drawn from a normal distribution with mean θ_{l_n} and variance ω_{l_n} (Equations 7-8). Defining h_n in this way makes the temporal distribution smooth, allowing us to fill in periods with only a few observations. However, we sacrifice some flexibility with this assumption, i.e., it does not capture multimodalities in presence for a single location l_n.

The conditional independence assumptions between the random variables are visually represented in Figure 2. Direct inference of all the parameters from the data is not possible in this model, requiring us to either optimise them (i.e., variational approximation) or to perform Markov chain Monte Carlo sampling [1]. Several effective and conceptually simple Gibbs sampling schemes are available for inference with a Dirichlet process, so we used the latter approach adapted from [16]. After obtaining samples (following convergence of the Markov chain), we can find the predictive distribution for location v given the entire training set $\boldsymbol{X}$ for each individual [1]:

$$pr(v|t_v, \boldsymbol{X}) = \frac{1}{R}\sum_{r=1}^{R} pr(v|t_v, \boldsymbol{M^{(r)}})pr(\boldsymbol{M^{(r)}}|\boldsymbol{X}) \tag{9}$$

where r is the index of each sample (taken after convergence), t_v is the query time, $\boldsymbol{M^{(r)}}$ is the entire set of model parameters found in sample r, and R is the total number of samples.

To test our approaches to optimisation and learning, we next apply them to the real cell tower observations.

4 EXPERIMENTAL RESULTS

In this section we use the real world cell tower mobility data of 50,000 people living in Ivory Coast, measured over 2 weeks, to assess the feasibility of crowdsourcing package delivery in Section 4.1. Then, using the same data, we evaluate our approach to prediction in Section 4.2, and optimisation under uncertainty in Section 4.3.

4.1 FEASIBILITY STUDY

To assess the feasibility of the idea of crowdsourcing package delivery, we consider three key criteria: (1) the number of participants required for acceptable geographical coverage; (2) the number of participants required in any specific delivery (since longer chains imply greater risk of loss and theft); and (3) the feasibility of delivering to rural locations, which is expected to be much harder than urban delivery. To assess these criteria, it was sufficient to consider a simpler instantiation (in this section only) of the problem we

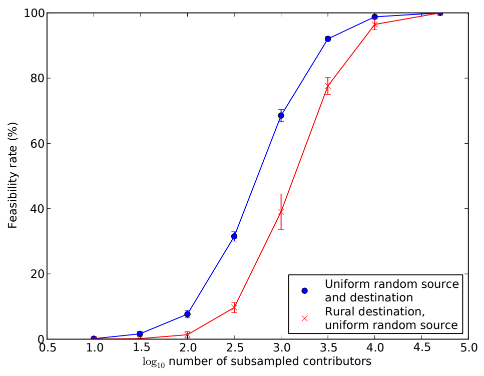


Figure 3: A plot of the percentage of randomly sampled (source,destination) delivery problems that had a solution path of any size, against the $\log_{10}$ size of the number of potential contributors.

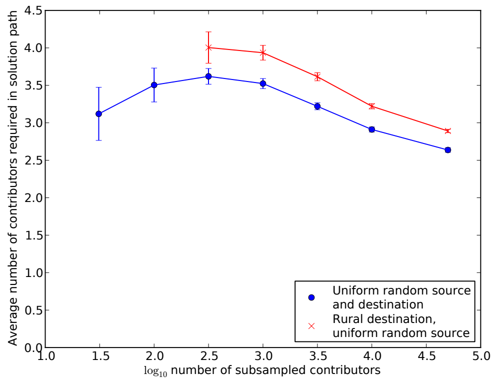


Figure 4: A plot of the average number of contributors required to each specific delivery problem (drawn from the much larger pool of potential contributors) against the $\log_{10}$ size of the potential contributors pool. N.B., a majority of rural destinations are infeasible for pool sizes of less than $10^{2.5}$, therefore we are unable to plot the line below this range.

defined in Section 3.1 that takes into account the locations that each person in the participant set, P, visited, but does not include the temporal structure in the mobility. We consider the temporal aspect of feasibility in Section 4.3.

4.1.1 Criterion 1: Number of Participants Required

To assess the number of participants required for wide geographical coverage (Criterion 1), we uniformly randomly subsampled participant sets, P', from the global participant set P (containing 50,000 people), for a wide range of different sizes $|P'| = \{10^{0.5i} | i = 1, 2, ..., 9\}$. For each participant set, we then uniformly sampled 1,000 pairs of locations (source and destination) from V representing 1,000 possible delivery problems. We consider a different (urban to rural) distribution of test locations in Section 4.1.3.

For each test location pair, we used Dijkstra's algorithm to find the shortest path (the standard algorithm can be applied to graph $\mathcal{G}$ because these is no uncertainty about the edge costs). Figure 3 shows the percentage of location pairs that were feasible (i.e., that had any path between the source and destination locations). The line with circular points shows the feasibility for uniform random source and destination locations. We see that the geographical coverage is very poor when there are fewer than $10^{2.5}$ participants. The critical range is around 10^3, when feasibility surges with each new participant. The heavy tail in human location behaviour is one explanation for this effect, where individuals visit many locations a few times (and a few locations many times) in their daily life mobility [8]. Therefore, an acceptable geographic coverage, trading off against recruitment/administration costs, appears to be around $10^{3.5}$ participants.

4.1.2 Criterion 2: Number of Participants Required for Any Given Delivery Problem

To assess the number of participants required in any given solution path (Criterion 2), we used the same subsampled participant sets as in Section 4.1.1 and plotted the length of the shortest path against the size of each subsampled participant set in Figure 4. The length of the shortest path indicates how many people are required for any specific delivery problem. The circular points are the focus for Criterion 2, where we see that the number of participants required for any solution path stays within the small range of 2 to 4. Since infeasible paths cannot be included when plotting Figure 4 (because they have unspecified numbers of contributors), the number of contributors required for specific paths initially increases with the size of the participant subset, as more paths are made feasible. However, once path feasibility (indicated in Figure 3) goes beyond 20%, the trend is as expected; having a wider pool of participants allows more efficient (i.e., shorter length) paths to be discovered. Note that, since we are not considering duration in Figure 4, the lowest cost paths in the full model may require more people. In any case, since the cost for losing the package can be fully specified by the planner, the optimal tradeoff between path length and duration can be found.

4.1.3 Criterion 3: Rural Distribution

So far, we have only considered uniformly sampled source and destination test points, which favours urban locations (since there are greater numbers of cell towers in urban areas). We now consider Criterion 3 for rural feasibility, by sampling a set of delivery problems where the destinations are only rural (keeping source locations uniformly sampled, as before).

Table 1: Average $\log_e$ data likelihood (higher is better) of held out test data of 50,000 individuals. 95% confidence intervals are given.

MODEL	LOG LIKELIHOOD
Our approach	-5.890 ± 0.057
First-order MM	-6.110 ± 0.043
VMM order 2 (Song et al. 2006)	-6.276 ± 0.030
VMM order 3	-6.347 ± 0.033
Random	-6.696 ± 0.056
Finite mixture, (Cho et al. 2011)	-9.452 ± 0.066

We ran the same analyses for Criteria 1 and 2 with rural destinations, yielding the lines with crosses in Figures 3 and 4. We conclude that restricting the destinations to be rural certainly makes the delivery problem more challenging, but it is still feasible. Now that we know that all three feasibility criteria are met, we consider the problem of learning the temporal structure in mobility to enable the minimisation of delay in delivery from source to destination nodes.

4.2 EVALUATION OF HUMAN MOBILITY PREDICTIONS

In this section, we evaluate our approach to predicting human mobility under considerable data sparsity, as would be typical from cell tower datasets. We split the cell tower data of 50,000 people into training and testing sets. The test set contains a single cell tower location reading from each person's data, therefore giving a test set of 50,000 data points. The rest of the data for the same individuals was used in training. To test for model quality, we looked at the logarithm of the data likelihood of each test point. We used non-informative hyperparameters $a = 1, d = 1, \alpha = 1$ for the discrete priors (see Figure 2). We used $b = (0.01, 12)$ and $c = (0.01, 3)$ for the continuous temporal priors, referring to the relative mean of precision w.r.t. the data, the mean of the prior, the degree of freedom in the precision, and the inverse mean of precision, respectively.

For comparison on the same data, we also tested two existing approaches that are considered state-of-the-art for human routine location prediction. The first is a spatio-temporal approach by Cho et al. (2011) [4], and the second method is a sequential approach by Song et al. (2006) [25] based on variable-order Markov models (VMM). In addition, we also tested a purely random model, with data likelihood $pr(x, d, h) = \frac{1}{L}\frac{1}{V}\mathcal{N}(h|\mu = 12, \sigma = 6)$, where L is the total number of locations and $V = 7$ is the number of days of the week (and (x, d, h) is the location, day of week, and hour of day observation as before).

The held-out data likelihood of all the approaches on the

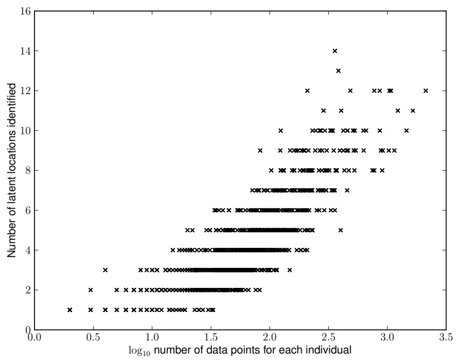


Figure 5: The number of individual latent locations identified by the Dirichlet process for 1,000 randomly selected individuals in the dataset.

50,000 data points can be seen in Table 1. We first note that VMM is worse than even a first-order Markov model, which is contrary to the findings of Song et al. (2006). This difference is due to the fact that the training data is very sparse, so learning higher-order dependencies causes a degradation in likelihood, even though the motivation behind fall-back (in the VMM) is to dynamically use orders appropriate to the context. Consequently, we see a further degradation as we increase the maximum order of the VMM to 3. We can also see that the model of Cho et al. (2011) performs the worst out of all the approaches. In contrast, our model outperforms all the others by at least 25% (since we are using a $\log_e$ likelihood). We believe most of this benefit comes from selecting the right number of components using the Dirichlet process (to let the data "speak for itself"). In Figure 5, we show the number of components (i.e., latent locations) found for a random subsample of 1,000 individuals, plotted against the dataset size for each individual. The number of latent locations has mean 4.1, mode 2, and standard deviation 2.4, with a heavy tail. Therefore, the bimodal assumption of Cho et al. (2011) is true for a large number of individuals in our dataset, yet, there are still many other individuals for whom their model is too complex, or not complex enough. Performance for these individuals that makes their model worse overall.

4.3 EVALUATION OF OPTIMISATION APPROACH

We now evaluate the optimisation element of our work (i.e., which participants to ask and which intermediate locations to use). To do this, we make a few additional assumptions in light of the results we have presented so far. Firstly, since a participant pool of approximately 3,500 people is enough to get satisfactory coverage of Ivory Coast (see Section 4.1), we used participant sets of this size in our optimisation evaluation. Secondly, in order to get statistically significant results, we ran 10,000 simulations using our mobility model (given in Section 3.2) as the ground truth, since it

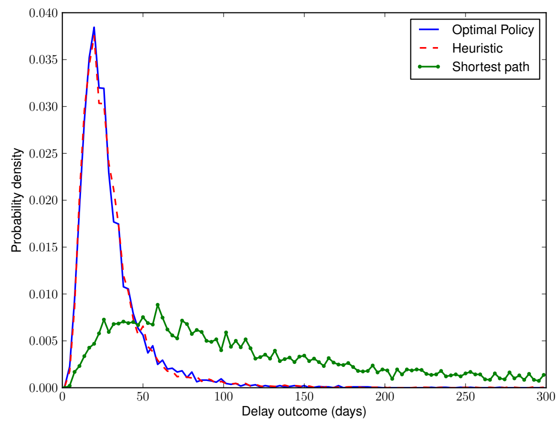


Figure 6: Probability distribution of delay cost in 10,000 simulated journeys to rural destinations using our MDP formulation (and heuristic) versus just taking the shortest path.

performs best under extreme data sparsity. To evaluate the robustness of the optimisation to uncertainties in human behaviour, we considered the total delay cost of 10,000 simulations, using the approach presented in Section 3.1 (using modified policy iteration to find the optimal policy [20]). To put this result in context, we also evaluated two alternative approaches to package routing as benchmarks. The first benchmark is the naïve approach that finds the shortest path (i.e., the minimum number of contributors), but does not consider the temporal mobility habits of the participants. The second benchmark uses the findings presented in Section 3.1, but finds the path of lowest expected cost during the planning stage instead of during runtime (and is therefore a heuristic based on our MDP formulation). This results in a policy for participant selection (i.e., who to ask to deliver the package given the time slot) but a *fixed* route. We therefore expect this approach to perform worse compared to the full optimal policy, since it is not able to react to optimally to incoming delay information.

The results are presented in Figure 6, showing the end-to-end duration performance of our optimal policy and heuristic approaches against the shortest path benchmark. For this, we used the rural test set, defined in Section 4.1.3, with an average of 373 km between the source and destination locations. The average total duration for the optimal policy is 30.0 days, versus 161 days for the benchmark. The heuristic we based our MDP formulation on performed almost identically to the optimal policy, with an average of 30.7 days duration. Interestingly, all three distributions are heavy-tailed, which conforms to expectations from other findings about delays from human behaviour [8]. There is, therefore, an 81.3% time advantage to learning and optimising over human behaviour, and it seems that without a consideration of the mobility habits of the participants, there would be an infeasible delay. Furthermore, since our heuristic performs almost as well as the optimal policy, there appears to be little benefit to being able to dynamically (at runtime) change the next location in response to the delays observed so far.

4.4 DISCUSSION

To perform routing under uncertainty, we assumed that the participants would follow their normal mobility patterns when delivering packages (see Section 3.1). Clearly, additional factors could introduce further delay, including disruptions to transport and short term disruptions arising from participants' circumstances (e.g., being too busy, taking sick leave). In practical terms, most of the impact of these disruptions could be absorbed by an appropriate task assignment procedure. Specifically, after obtaining a policy from our learning and optimisation approach, the system could ask the selected participants, via automated phone text, whether they are actually willing and able to do the task. In this way, participants facing disruptions can be filtered out, limiting the introduction of unexpected delay into the route. On the other hand, some disruptions may not be known at the time of task acceptance, or some participants may simply not be honest about them. We leave this as a problem for future work (see Section 5).

Finally, in the worst case (from a routing perspective), participants may lose or steal packages. A certain amount of loss and theft is assumed even with standard delivery, and is borne as the risk of doing business, or addressed with insurance. In the crowdsourced setting, this can be taken into account by assigning a cost to each participant (either with a fixed value, or derived from a participant-specific trust evaluation framework). In whatever way the cost of trust is calculated, once obtained, it can be incorporated into the MDP as an added cost in the standard way.

5 CONCLUSIONS AND FUTURE WORK

In this work we studied a novel method for distribution that uses the existing mobility of local people to send packages large distances. Using data describing the real world movement patterns of 50,000 people, we addressed the technical problems associated with this method, formulating an MDP for optimisation and presenting a Bayesian non-parametric model that performs well under data sparsity. Future work could incorporate the most recent observations of participants' locations in order to respond to unexpected delays (in addition to the random variability in delay attributable to daily life mobility that we already did consider). Introducing this sequential dependence breaks the periodic feature of the predictions, making the MDP intractable again. To address this, a hybrid approach could be developed that assumes periodicity during initial planning, but which allows local refinements to the policy as up-to-the-hour information about participant mobility arrives.

References

[1] C. M. Bishop. *Pattern recognition and machine learning*, volume 4. Springer New York, 2006.

[2] J. Chen, K. H. Low, C. K.-Y. Tan, A. Oran, P. Jaillet, J. Dolan, and G. Sukhatme. Decentralized data fusion and active sensing with mobile sensors for modeling and predicting spatiotemporal traffic phenomena. In *Proceedings of the Twenty-Eighth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-12)*, pages 163–173, Corvallis, Oregon, 2012.

[3] M. Cherubini, M. Zhu, N. Oliver, and M. Cebrian. Exploring Social Networks as an Infrastructure for Transportation Networks. In *Presentation at the International School and Conference on Network Science (NetSci'10)*, Boston, MA, USA, 2010.

[4] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1082–1090. ACM, 2011.

[5] CIA Directorate of Intelligence. The world factbook. July 2008.

[6] N. Eagle and A. S. Pentland. Eigenbehaviors: identifying structure in routine. *Behavioral Ecology and Sociobiology*, 63(7):1057–1066, 2009.

[7] H. Gao, J. Tang, and H. Liu. Exploring social-historical ties on location-based social networks. In *6th International AAAI Conference on Weblogs and Social Media*, 2012.

[8] M. Gonzalez, C. Hidalgo, and A. Barabasi. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, June 2008.

[9] L. Hufnagel, D. Brockmann, and T. Geisel. Forecast and control of epidemics in a globalized world. *Proceedings of the National Academy of Sciences of the United States of America*, 101(42):15124–15129, 2004.

[10] B. Keller, P. von Bergen, R. Wattenhofer, and S. Welten. On the feasibility of opportunistic ad hoc music sharing. *Mobile Data Challenge by Nokia Workshop, in conjunction with International Conference on Pervasive Computing*, 2012.

[11] K. Laskey, N. Xu, and C. H. Chen. Propagation of delays in the national airspace system. In *Proceedings of the Twenty-Second Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, pages 265–272, Arlington, Virginia, 2006.

[12] J. Laurila, D. Gatica-Perez, I. Aad, J. Blom, O. Bornet, T. Do, O. Dousse, J. Eberle, and M. Miettinen. The mobile data challenge: Big data for mobile computing research. In *Mobile Data Challenge by Nokia Workshop, in conjunction with International Conference on Pervasive Computing*, Newcastle, UK, 2012.

[13] C. Liu and J. Wu. Practical routing in a cyclic mobispace. *Networking, IEEE/ACM Transactions on*, 19(2):369–382, 2011.

[14] J. McInerney, A. Rogers, and N. R. Jennings. Improving location prediction services for new users with probabilistic latent semantic analysis. *In Mobile Data Challenge by Nokia Workshop, in conjunction with International Conference on Pervasive Computing*, 2012.

[15] J. McInerney, J. Zheng, A. Rogers, and N. R. Jennings. Modelling heterogeneous location habits in human populations for location prediction under data sparsity. In *International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2013)*, in press.

[16] R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2):249–265, 2000.

[17] E. Nikolova, M. Brand, and D. R. Karger. Optimal route planning under uncertainty. In *Proceedings of International Conference on Automated Planning and Scheduling*, 2006.

[18] E. Nikolova and D. R. Karger. Route planning under uncertainty: The canadian traveller problem. In *Proc. AAAI*, pages 969–974, 2008.

[19] G. Pickard, I. Rahwan, W. Pan, M. Cebrian, R. Crane, A. Madan, and A. Pentland. Time critical social mobilization: The darpa network challenge winning strategy. 2010.

[20] M. L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc., 1994.

[21] I. Rahwan, S. Dsouza, A. Rutherford, V. Naroditskiy, J. McInerney, M. Venanzi, N. Jennings, and M. Cebrian. Global manhunt pushes the limits of social mobilization. *IEEE Computer*, 46(4):68–75, 2013.

[22] A. Sadilek and J. Krumm. Far out: Predicting long-term human mobility. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[23] S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. Campbell. Nextplace: a spatio-temporal prediction framework for pervasive systems. In *Pervasive Computing*, pages 152–169, San Francisco, CA, USA, 2011. Springer.

[24] J. Scott, A. J. Brush, J. Krumm, B. Meyers, M. Hazas, S. Hodges, and N. Villar. PreHeat: controlling home heating using occupancy prediction. In *Proceedings of the 13th international conference on Ubiquitous computing (UbiComp 2011)*, pages 281–290, Beijing, China, 2011.

[25] L. Song, D. Kotz, R. Jain, and X. He. Evaluating next-cell predictors with extensive wi-fi mobility data. *IEEE Transactions on Mobile Computing*, 5(12):1633–1649, 2006.

[26] E. Stevens-Navarro, Y. Lin, and V. W. Wong. An mdp-based vertical handoff decision algorithm for heterogeneous wireless networks. *Vehicular Technology, IEEE Transactions on*, 57(2):1243–1254, 2008.

[27] V. Vukadinović, Ó. R. Helgason, and G. Karlsson. A mobility model for pedestrian content distribution. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, page 93. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009.

[28] A. Wesolowski, N. Eagle, A. J. Tatem, D. L. Smith, A. M. Noor, R. W. Snow, and C. O. Buckee. Quantifying the impact of human mobility on malaria. *Science*, 338(6104):267–270, 2012.

[29] J. H. Wu and R. Givan. Feature-discovering approximate value iteration methods. *Abstraction, Reformulation and Approximation*, pages 901–901, 2005.

Learning Max-Margin Tree Predictors

Ofer Meshi[†*] **Elad Eban**[†*] **Gal Elidan**[‡] **Amir Globerson**[†]

[†] School of Computer Science and Engineering
[‡] Department of Statistics
The Hebrew University of Jerusalem, Israel

Abstract

Structured prediction is a powerful framework for coping with joint prediction of interacting outputs. A central difficulty in using this framework is that often the correct label dependence structure is unknown. At the same time, we would like to avoid an overly complex structure that will lead to intractable prediction. In this work we address the challenge of learning tree structured predictive models that achieve high accuracy while at the same time facilitate efficient (linear time) inference. We start by proving that this task is in general NP-hard, and then suggest an approximate alternative. Our CRANK approach relies on a novel Circuit-RANK regularizer that penalizes non-tree structures and can be optimized using a convex-concave procedure. We demonstrate the effectiveness of our approach on several domains and show that its accuracy matches that of fully connected models, while performing prediction substantially faster.

1 Introduction

Numerous applications involve joint prediction of complex outputs. For example, in document classification the goal is to assign the most relevant (possibly multiple) topics to each document; in gene annotation, we would like to assign each gene a set of relevant functional tags out of a large set of possible cellular functions; in medical diagnosis, we would like to identify all the diseases a given patient suffers from. Although the output space in such problems is typically very large, it often has intrinsic *structure* which can be exploited to construct efficient predictors. Indeed, in recent years using *structured output prediction* has resulted in state-of-the-art results in many real-worlds problems from computer vision, natural language processing, computational biology, and other fields [Bakir et al., 2007]. Such predictors can be learned from data using formulations such as Max-Margin Markov Networks (M^3N) [Taskar et al., 2003, Tsochantaridis et al., 2006], or conditional random fields (CRF) [Lafferty et al., 2001].

While the prediction and the learning tasks are generally computationally intractable [Shimony, 1994, Sontag et al., 2010], for some models they can be carried out efficiently. For example, when the model consists of pairwise dependencies between output variables, and these form a tree structure, prediction can be computed efficiently using dynamic programming at a linear cost in the number of output variables [Pearl, 1988]. Moreover, despite their simplicity, tree structured models are often sufficiently expressive to yield highly accurate predictors. Accordingly, much of the research on structured prediction focused on this setting [e.g., Lafferty et al., 2001, Collins, 2002, Taskar et al., 2003, Tsochantaridis et al., 2006].

Given the above success of tree structured models, it is unfortunate that in many scenarios, such as a document classification task, there is no obvious way in which to choose the most beneficial tree. Thus, a natural question is how to find the tree model that best fits a given structured prediction problem. This is precisely the problem we address in the current paper. Specifically, we ask what is the tree structure that is optimal in terms of a max-margin objective [Taskar et al., 2003]. Somewhat surprisingly, this optimal tree problem has received very little attention in the context of discriminative structured prediction (the most relevant work is Bradley and Guestrin [2010] which we address in Section 6).

Our contributions are as follows. We begin by proving that it is NP-hard in general to find the optimal max-margin predictive tree, in marked contrast to

*Authors contributed equally.

the generative case where the optimal tree can be learned efficiently [Chow and Liu, 1968]. To cope with this theoretical barrier, we propose an approximation scheme that uses regularization to penalize non-tree models. Concretely, we propose a regularizer that is based on the *circuit rank* of a graph [Berge, 1962], namely the minimal number of edges that need to be removed from the graph in order to obtain a tree. Minimization of the resulting objective is still difficult, and we further approximate it using a difference of continuous convex envelopes. The resulting objective can then be readily optimized using the convex concave procedure [Yuille and Rangarajan, 2003].

We apply our method to synthetic and varied real-world structured output prediction tasks. First, we show that the learned tree model is competitive with a fully connected max-margin model that is substantially more computationally demanding at prediction time. Second, we show that our approach is superior to several baseline alternatives (e.g., greedy structure learning) in terms of generalization performance and running time.

2 The Max-margin Tree

Let x be an input vector (e.g., a document) and y a discrete output vector (e.g., topics assigned to the document, where $y_i = 1$ when topic i is addressed in x). As in most structured prediction approaches, we assume that inputs are mapped to outputs according to a linear discrimination rule: $y(x; w) = \operatorname{argmax}_{y'} w^\top \phi(x, y')$, where $\phi(x, y)$ is a function that maps input-output pairs to a feature vector, and w is the corresponding weight vector. We will call $w^\top \phi(x, y')$ the *score* that is assigned to the prediction y' given an input x.

Assume we have a set of M labeled pairs $\{(x^m, y^m)\}_{m=1}^M$, and would like to learn w. In the M^3N formulation proposed by Taskar et al. [2003], w is learned by minimizing the following (regularized) structured hinge loss:

$$\ell(w) = \frac{\lambda}{2}\|w\|^2 + \frac{1}{M}\sum_m h^m(w),$$

where

$$h^m(w) = \max_y \left[w^\top \phi(x^m, y) + \Delta(y, y^m)\right] - w^\top \phi(x^m, y^m), \tag{1}$$

and $\Delta(y, y^m)$ is a label-loss function measuring the cost of predicting y when the true label is y^m (e.g., 0/1 or Hamming distance). Thus, the learning problem involves a loss-augmented prediction problem for each training example.

Since the space of possible outputs may be quite large, maximization of y can be computationally intractable. It is therefore useful to consider score functions that decompose into simpler ones. One such decomposition that is commonly used consists of scores over single variables and pairs of variables that correspond to nodes and edges of a graph G, respectively:

$$w^\top \phi(x, y) = \sum_{ij \in E(G)} w_{ij}^\top \phi_{ij}(x, y_i, y_j) + \sum_{i \in V(G)} w_i^\top \phi_i(x, y_i). \tag{2}$$

Importantly, when the graph G has a tree structure then the maximization over y can be solved exactly and efficiently using dynamic programming algorithms (e.g., Belief Propagation [Pearl, 1988]).

As mentioned above, we consider problems where there is no natural way to choose a particular tree structure, and our goal is to learn the optimal tree from training data. We next formalize this objective.

In a tree structured model, the set of edges ij in Eq. (2) forms a tree. This is equivalent to requiring that the vectors w_{ij} in Eq. (2) be non-zero only on edges of some tree. To make this precise, we first define, for a *given* spanning tree T, the set $\mathcal{W}_T$ of weight vectors that "agree" with T:[1]

$$\mathcal{W}_T = \{w : \; ij \notin T \implies w_{ij} = 0\}. \tag{3}$$

Next we consider the set $\mathcal{W}_\cup$ of weight vectors that agree with *some* spanning tree. Denote the set of all spanning trees by $\mathcal{T}$, then: $\mathcal{W}_\cup = \bigcup_{T \in \mathcal{T}} \mathcal{W}_T$. The problem of finding the optimal max-margin tree predictor is therefore:

$$\min_{w \in \mathcal{W}_\cup} \ell(w). \tag{4}$$

We denote this as the $M^{Tree}N$ problem. In what follows, we first show that this problem is NP-hard, and then present an approximation scheme.

3 Learning M^3N Trees is NP-hard

We start by showing that learning the optimal tree in the discriminative max-margin setting is NP-hard. As noted, this is somewhat of a surprise given that the best tree is easily learned in the generative setting [Chow and Liu, 1968], and that tree structured models are often used due to their computational advantages.

In particular, we consider the problem of deciding whether there exists a tree structured model that correctly labels a given dataset (i.e., deciding whether the

[1]Note that weights corresponding to single node features are not restricted.

dataset is separable with a tree model). Formally, we define the $M^{Tree}N$ *decision problem* as determining whether the following set is empty:

$$\Big\{w \in \mathcal{W}_\cup \Big| w^\top \phi(x^m, y^m) \geq w^\top \phi(x^m, y) + \Delta(y, y^m)\ \forall m, y\Big\}. \tag{5}$$

To facilitate the identifiability of the model parameters that is later needed, we adopt the formalism of Sontag et al. [2010] and define the score:

$$\begin{aligned} & S(y; x, T, w) \\ &= \sum_{ij\in T} w_{ij}^\top \phi_{ij}(x, y_i, y_j) + \sum_i (w_i^\top \phi_i(x, y_i) + x_i(y_i)) \\ &\equiv \sum_{ij\in T} \theta_{ij}(y_i, y_j) + \sum_i \theta_i(y_i) + \sum_i x_i(y_i), \end{aligned} \tag{6}$$

where $x_i(y_i)$ is a bias term which does not depend on w,[2] and for notational convenience we have dropped the dependence of θ on x and w. We can now reformulate the set in Eq. (5) as:

$$\left\{T, w \Big| S(y^m; x^m, T, w) \geq \max_y S(y; x^m, T, w)\ \forall m \right\}, \tag{7}$$

where, for simplicity, we omit the label loss $\Delta(y, y^m)$. This is valid since the bias terms already ensure that the trivial solution $w = 0$ is avoided. With this reformulation, we can now state the hardness result.

Theorem 3.1. *Given a set of training examples* $\{(x^m, y^m)\}_{m=1}^M$*, it is NP-hard to decide whether there exists a tree* T *and weights* w *such that* $\forall m, y^m = \operatorname{argmax}_y S(y; x^m, T, w)$.

Proof. We show a reduction from the NP-hard *bounded-degree spanning tree* (BDST) problem to the $M^{Tree}N$ decision problem defined in Eq. (7).[3] In the BDST problem, given an undirected graph G and an integer D, the goal is to decide whether there exists a spanning tree with maximum degree D (for $D = 2$ this is the Hamiltonian path problem, hence the NP-hardness).

Given an instance of BDST we construct an instance of problem Eq. (7) on the same graph G as follows. First, we define variables $y_1, \ldots, y_n$ that take values in $\{0, 1, \ldots, n\}$, where $n = |V(G)|$. Second, we will define the parameters $\theta_i(y_i)$ and $\theta_{ij}(y_i, y_j)$ and bias terms $x_i(y_i)$ in such a way that solving the $M^{Tree}N$ decision problem will also solve the BDST problem. To complement this, we will define a set of training examples which are separable only by the desired parameters. For clarity of exposition, we defer the proof that these parameters are identifiable using a polynomial number of training examples to App. A.

[2]As usual, the bias term can be removed by fixing some elements of w to 1.

[3]A related reduction is given in Aissi et al. [2005], Theorem 8.

The singleton parameters are

$$\theta_i(y_i) = \begin{cases} D & i = 1, y_1 = 0 \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

and the pairwise parameters for $ij \in E(G)$ are:

$$\theta_{ij}(y_i, y_j) = \begin{cases} -n^2 & y_i \neq y_j \\ 0 & y_i = y_j = 0 \\ 1 & y_i = y_j = i \\ 1 & y_i = y_j = j \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

Now consider the case where the bias term $x_i(y_i)$ is identically zero. In this case the score for (non-zero) uniform assignments equals the degree of the vertex in T. That is, $S(i, \ldots, i; 0, T, \theta) = \deg_T(i)$ since $\theta_{ij}(i, i) = 1$ for all $j \in N(i)$ and the other parameter values are zero. The score for the assignment $y = 0$ is $S(0, \ldots, 0; 0, T, \theta) = D$, and for non-uniform assignments we get a negative score $S(y; 0, T, \theta) < 0$. Therefore, the maximization over y in Eq. (7) reduces to a maximization over n uniform states (each corresponding to a vertex in the graph). The maximum value is thus the maximum between D and the maximum degree in T: $\max_y S(y; 0, T, \theta) = \max\{D, \max_i \deg_T(i)\}$.

It follows that, if we augment the training set that realizes the above parameters (see App. A) with a single training example where $x^m = y^m = (0, \ldots, 0)$, the set of Eq. (7) can now be written as:

$$\left\{T | D \geq \max_i \deg_T(i)\right\}.$$

Thus, we have that the learning problem is separable if and only if there exists a bounded degree spanning tree in G. This concludes the reduction, and we have shown that the decision problem in Theorem 3.1 is indeed NP-hard. □

The above hardness proof illustrates a striking difference between generative learning of tree models (i.e., Chow Liu) and discriminative learning (our NP-hard setting). Clearly, we do not want to abandon the discriminative setting and trees remain computationally appealing at test time. Thus, in the rest of the paper, we propose practical approaches for learning a tree structured predictive model and demonstrate empirically that our method is competitive.

4 Tree-Inducing Regularization

Due to the hardness of the tree learning problem, we next develop an approximation scheme for it. We begin with the exact formulation of the problem, and then introduce an approximate formulation along with an optimization procedure. Our construction relies on several properties of submodular functions and their convex envelopes.

4.1 Exact Tree Regularization

As described in Section 2, we would like to find a tree structured weight vector $w \in \mathcal{W}_{\cup}$ that minimizes the empirical hinge loss. The main difficulty in doing so is that the sparsity pattern of w needs to obey a fairly complex constraint, namely being tree structured. This is in marked contrast to popular sparsity constraints such as an upper bound on the number of non-zero values, a constraint that does not take into account the resulting global structure.

To overcome this difficulty, we will formulate the exact learning problem via an appropriate regularization. We begin by defining a function that maps w to the space of edges: $\pi : \mathbb{R}^d \mapsto \mathbb{R}^{|E|}$, where E corresponds to all edges of the full graph. Specifically, the component in π corresponding to the edge ij is:

$$\pi_{ij}(w) = \|w_{ij}\|_1.$$

Now, denote by $Supp(\pi(w))$ the set of coordinates in $\pi(w)$ that are non-zero. We would like the edges corresponding to these coordinates to form a tree graph. Thus, we wish to define a set function $F(Supp(\pi(w)))$ which will be equal to zero if $Supp(\pi(w))$ conforms to *some* tree structure, and a positive value otherwise. If we then add $\beta F(Supp(\pi(w)))$ to the objective in Eq. (4) with β large enough, the resulting w will be tree structured. The optimization problem is then:

$$\min_w \ell(w) + \beta F(Supp(\pi(w))). \tag{10}$$

In what follows, we define a set function $F(A)$ with the desired properties. That is, we seek a function that takes a set of edges A and outputs zero if they correspond to a tree, and a positive number otherwise. Intuitively, it is also desirable to define the function such that its value increases as the graph becomes less "tree-like". To make this concrete we define a measure for "treeness" as the minimum number of edges that need to be removed from A in order to reduce it to a tree structure. This measure is also known as the *circuit-rank* of the graph [Berge, 1962]. Formally:

$$r = |A| + c(A) - n,$$

where $c(A)$ is the number of connected components in the graph, and n is the number of vertices. We note that the circuit rank is also the co-rank of the graphic matroid, and is hence supermodular.

Putting it all together, we have that our desired tree-inducing function is given by:

$$F(Supp(\pi(w))) = |Supp(\pi(w))| + c(Supp(\pi(w))) - n.$$

Of course, given our hardness result, optimizing Eq. (10) with the above $F(A)$ is still computationally hard. From an optimization perspective, the difficulty comes from the non-convexity of the the above function in w. Optimization is further complicated by the fact that it is highly non-smooth, similarly to the ℓ_0 norm. In the next section we suggest a smoothed approximation that is more amenable to optimization.

We also note that in Eq. (10) w is generally not tree structured, so the maximization over y in Eq. (1) is not necessarily tractable. Therefore, we replace the hinge loss in Eq. (1) with its *overgenerating* approximation [Finley and Joachims, 2008], known as linear programming (LP) relaxation [e.g., see Meshi et al., 2010]. This is achieved by formulating the optimization in Eq. (1) as an integer LP and then relaxing the integrality requirement, allowing fractional solutions. Importantly, for tree structured graphs this approximation is in fact exact [Wainwright and Jordan, 2008]. This implies that if there exists a tree model that separates the data, it will be found even when using this relaxation in Eq. (10). With slight abuse of notation, we keep referring to the approximate objective as $\ell(w)$.

4.2 Approximate Tree Regularization

The function $F(A)$ is a set function applied to the support of $\pi(w)$. Bach [2010] (Proposition 1) shows that when F is submodular *and* non-decreasing, the convex envelope of $F(Supp(\pi(w)))$ can be calculated efficiently. This is desirable since the convex envelope then serves as a convex regularizer. Furthermore, this convex envelope can be elegantly understood as the Lovász extension f of F, applied to $|\pi(w)|$ (in our case, $|\pi(w)| = \pi(w)$). Unfortunately, the circuit-rank r does not satisfy these conditions, since it is supermodular and non-decreasing (in fact, its convex envelope is a constant).

To overcome this difficulty, we observe that $F(A)$ can be decomposed in a way that allows us to use the result of Bach [2010]. Specifically, we can write $F(A) = F_1(A) - F_2(A)$, where

$$F_1(A) = |A| \quad , \quad F_2(A) = n - c(A). \tag{11}$$

$F_1(A)$ is simply the cardinality function which is modular and increasing. Furthermore, $F_2(A)$ is the rank of the graphic matroid [Oxley, 2006], and is hence submodular. It is also easy to see that $F_2(A)$ is non-decreasing. Thus, both functions satisfy the conditions of Proposition 1 in Bach [2010] and their convex envelopes can be found in closed form, as characterized in the following corollaries.

Corollary 4.1. *The convex envelope of $F_1(Supp(\pi(w))$ is $f_1(\pi(w)) = \sum_{ij} \pi_{ij}(w) = \|w\|_1$.*

Proof. Follows directly from Prop. 1 in Bach [2010] and the fact that the Lovász extension of the cardinality function is the ℓ_1 norm. $\square$

Corollary 4.2. *The convex envelope of $F_2(Supp(\pi(w))$ is $f_2(\pi(w))$, defined as follows. Sort the elements of $\pi(w)$ in decreasing order, and construct a maximum-spanning-tree with this ordering as in Kruskal's algorithm [Kruskal, 1956]. Denoting the resulting tree by $T(\pi(w))$, we obtain*

$$f_2(\pi(w)) = \sum_{ij \in T(\pi(w))} \pi_{ij}(w) = \sum_{ij \in T(\pi(w))} \|w_{ij}\|_1$$

Proof. Let $(ij)_k$ denote the k^{th} edge when sorting $\pi(w)$, then the Lovász extension f_2 of F_2 at $\pi(w)$ is:

$$\sum_{k=1}^{|E|} \pi_{(ij)_k}(w)[F_2(\{(ij)_1, \dots, (ij)_k) - F_2(\{(ij)_1, \dots, (ij)_{k-1})]$$
$$= \sum_{ij \in T(\pi(w))} \pi_{ij}(w),$$

where we have used Eq. (11) and the fact that the number of connected components decreases by one only when introducing edges in Kruskal's tree. The desired result follows from Prop. 1 in [Bach, 2010]. $\square$

We now approximate $F(Supp(\pi(w)))$ as a difference of the two corresponding convex envelopes, denoted by $f(\pi(w))$:

$$f(\pi(w)) \equiv f_1(\pi(w)) - f_2(\pi(w)) = \sum_{ij \notin T(\pi(w))} \|w_{ij}\|_1 \tag{12}$$

This function has two properties that make it computationally and conceptually appealing:

- $f(\pi(w))$ is a difference of two convex functions so that a local minimum can be easily found using the convex concave procedure (see Section 4.3).
- The set $\{ij \notin T(\pi(w))\}$ are precisely these edges that form a cycle when added according to the order implied by $\pi(w)$. Thus, the penalty we use corresponds to the magnitude of $\|w_{ij}\|_1$ on the edges that form cycles, namely the non-tree edges.

4.3 Optimizing with Approximate Tree Regularization

Using the tree inducing regularizer from the previous section, our overall optimization problem becomes:

$$\min_w \ell(w) + \beta f_1(\pi(w)) - \beta f_2(\pi(w)).$$

Since the function $f_2(\pi(w))$ is rather elaborate, optimizing the above objective still requires care. In what follows, we introduce a simple procedure for doing so that utilizes the convex concave procedure (CCCP) [Yuille and Rangarajan, 2003].[4] Recall that CCCP is applicable for an objective function (to be minimized) that is a sum of a convex and concave functions, and proceeds via linearization of the concave part.

To use CCCP for our problem we observe that from the discussion of the previous section it follows that our objective can indeed be decomposed into the following convex and concave components:

$$h_\cup(w) = \ell(w) + \beta f_1(\pi(w)), \quad h_\cap(w) = -\beta f_2(\pi(w)),$$

where $\cap$ and $\cup$ correspond to the convex and concave parts, respectively. To linearize $h_\cap(w)$ around a point w^t, we need to find its subgradient at that point. The next proposition, which follows easily from Hazan and Kale [2009], gives the subgradient of $f_2(\pi(w))$:

Proposition 4.3. *The subgradient of $f_2(\pi(w))$ is given by the vector v defined as follows.[5] The coordinates in v corresponding to w_{ij} are given by:*

$$v_{ij} = \begin{cases} sign(w_{ij}) & ij \in T(\pi(w)) \\ 0 & otherwise \end{cases}$$

where sign is taken element wise. The other coordinates of v (corresponding to w_i) are zero.

We can now specify the resulting algorithm, which we call CRANK for circuit-rank regularizer.

Algorithm 1 The CRANK algorithm

```
Input: $w^1, \beta$
for $t = 1, \dots$ do
    $h^t(w) = \ell(w) + \beta\|w\|_1 - \beta v(w^t)^\top w$
    $w^{t+1} = \text{argmin}_w h^t(w)$
end for
```

The objective $h^t(w)$ to be minimized at each iteration is a convex function, which can be optimized using any convex optimization method. In this work we use the

[4]To be precise, we are using the more general DC programming framework [Tao and An, 1997], which can be applied to non differentiable functions.

[5]In cases where several w_{ij} are equal, there are multiple subgradients.

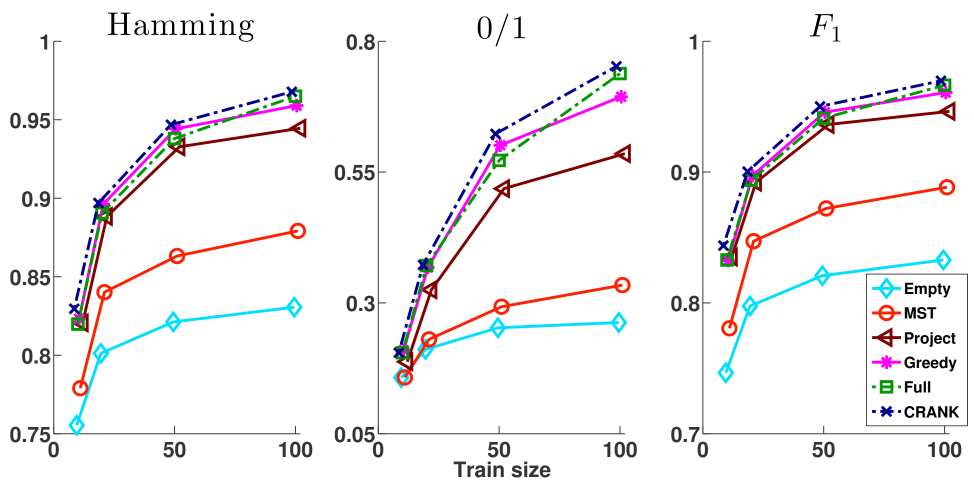


Figure 1: Average test performance as a function of the number of training samples for the synthetic datasets.

stochastic Frank-Wolfe procedure recently proposed by Lacoste-Julien et al. [2013].[6] The advantage of this approach is that the updates are simple, and it generates primal and dual bounds which help monitor convergence. In practice, we do not solve the inner optimization problems exactly, but rather up to some primal-dual gap.

5 Experiments

In this section we evaluate the proposed algorithm on multi-label classification tasks and compare its performance to several baselines. In this task the goal is to predict the subset of labels which best fits a given input. We use the model presented in Finley and Joachims [2008], where each possible label $y_i \in \{0,1\}$ is associated with a weight vector w_i, the singleton scores are given by $w_i^\top x y_i$, and the pairwise scores are simply $w_{ij} y_i y_j$ (i.e., w_{ij} is scalar).

We compare our **CRANK** algorithm to the following baselines: The **Empty** model learns each label prediction independently of the other labels; The **Full** model learns a model that can use all pairwise dependencies; The **Greedy** trainer starts with the empty model and at each iteration adds the edge which achieves the largest gain in objective while not forming a cycle, until no more edges can be added; The **Project** algorithm, runs CRANK starting from the weights learned by the Full algorithm, and using a large penalty β; [7] The final baseline is an **MST** algorithm, which calculates the gain in objective for each edge separately, takes a maximum-spanning-tree over these weights, and then re-trains the resulting tree. Since CCCP may be sensitive to its starting point, we restart CRANK from 10 random points and choose the one with lowest objective (we run those in parallel). We apply the stochastic Frank-Wolfe algorithm [Lacoste-Julien et al., 2013] to optimize the weights in all algorithms. The Full and CRANK algorithms operate on non-tree graphs, and thus use an LP relaxation within the training loss (see Section 4.1). Since the model trained with Full is not tree structured, we also needs to use LP relaxation at test time (see implications on runtime below). We used the GLPK solver for solving the LPs.

To measure the performance of the algorithms, we consider three accuracy measures: **Hamming**, **zero-one**, and $\boldsymbol{F_1}$ (averaged over samples). See [Zhang and Schneider, 2012, Dembczynski et al., 2010] for similar evaluation schemes. Regularization coefficients were chosen using cross-validation. The parameter β in CRANK was gradually increased until a tree structure was obtained.

Synthetic Data: We first show results for synthetic data where $x \in \mathbb{R}^4$. The data was created as follows: a random tree T over $n = 10$ variables was picked and corresponding weights $w \in \mathcal{W}_T$ were sampled. Train and test sets were generated randomly and labeled using w. Test set size was 1000 and train set size varied. Results (averaged over 10 repetitions) are shown in Figure 1.

We observe that the structured models do significantly better than the Empty model. Additionally, we see that the Full, Greedy, and CRANK algorithms are comparable in terms of prediction quality, with a slight advantage for CRANK over the others. We also notice that Project and MST do much worse than the other structured models.

[6]We modified the algorithm to handle the $\ell_1 + \ell_2$ case.

[7]The Project scheme thus trains a model consisting of the maximum-spanning-tree over the weights learned by Full, and can be viewed as a "tree-projection" of the full model.

	Hamming	0/1	F_1	Hamming	0/1	F_1
	Scene			Emotions		
CRANK	90.5 (2)	58.9 (2)	64.8 (2)	**79.2** (1)	29.2 (2)	60.5 (2)
Full	**90.7** (1)	**62.2** (1)	**67.8** (1)	79.0 (3)	**33.7** (1)	**62.5** (1)
Greedy	90.2 (3)	56.9 (3)	62.6 (3)	78.5 (4)	24.3 (4)	54.5 (4)
Project	89.5 (5)	52.1 (5)	59.2 (5)	77.6 (5)	20.8 (5)	49.8 (5)
MST	89.9 (4)	53.0 (4)	59.6 (4)	79.1 (2)	28.2 (3)	57.5 (3)
Empty	89.3 (6)	49.5 (6)	56.5 (6)	76.7 (6)	20.3 (6)	48.5 (6)
	Medical			Yeast		
CRANK	**96.9** (1)	74.0 (2)	78.2 (3)	80.1 (2)	17.6 (2)	60.4 (3)
Full	**96.9** (1)	**75.0** (1)	**78.3** (1)	**80.2** (1)	**19.0** (1)	**60.9** (1)
Greedy	**96.9** (1)	74.0 (2)	**78.3** (1)	NA	NA	NA
Project	96.7 (4)	72.7 (4)	77.0 (5)	80.1 (2)	16.4 (3)	60.7 (2)
MST	96.7 (4)	71.9 (5)	77.5 (4)	80.1 (2)	16.1 (4)	60.3 (4)
Empty	96.4 (6)	71.0 (6)	76.1 (6)	79.8 (5)	12.1 (5)	58.0 (5)

Table 1: Performance on test data for real-world multi-label datasets. The rank of each algorithm for each dataset and evaluation measure is shown in brackets. Greedy was too slow to run on Yeast.

Real Data: We next performed experiments on four real-world datasets.[8] In the **Scene** dataset the task is to classify a given image into several outdoor scene types (6 labels, 294 features). In the **Emotions** dataset we wish to assign emotional categories to musical tracks (6 labels, 72 features). In the **Medical** dataset the task is document classification (reduced to 10 labels, 1449 features). This is the experimental setting used by Zhang and Schneider [2012]. Finally, to test how training time scales with the number of labels, we also experiment with the **Yeast** dataset, where the goal is to predict which functional classes each gene belongs to (14 labels, 103 features). The results are summarized in Table 1.

We first observe that in this setting the Full model generally has the best performance and Empty the worst. As before, CRANK is typically close to Full and outperforms Greedy and the other baselines in the majority of the cases.

Runtime analysis: In Table 2 we report train and test run times, relative to the Full model, for the Yeast dataset.

Table 2: Running times for the Yeast dataset.

Time	CRANK	Full	Project	MST	Empty
Train	1.82	1.0	0.17	1.32	0.01
Test	0.02	1.0	0.06	0.02	0.02

It is important to note that the greedy algorithm is very slow to train, since it requires solving $O(n^3)$ training problems. It is thus impractical for large problems, and this is true even for the Yeast dataset which has only $n = 14$ labels (and hence does not appear in Table 2). The Full model on the other hand has much longer test times (compared to the tree-based models), since it must use an LP solver for prediction. CRANK has a training time that is reasonable (comparable to Full) and a much better test time. On the Yeast dataset prediction with CRANK is 50 times faster than with Full.

In conclusion, CRANK seems to strike the best balance in terms of accuracy, training time, and test time: it achieves an accuracy that is close to the best among the baselines, with a scalable training time, and very fast test time. This is particularly appealing for applications where we can afford to spend some more time on training while test time is critical (e.g., real-time systems).

6 Related Work

The problem of structure learning in graphical models has a long history, dating back to the celebrated algorithm by Chow and Liu [1968] for finding a maximum likelihood tree in a generative model. More recently, several elegant works have shown the utility of ℓ_1 regularization for structure learning in generative models [Friedman et al., 2008, Lee et al., 2007, Ravikumar et al., 2008, 2010]. These works involve various forms of ℓ_1 regularization on the model parameters, coupled with approximation of the likelihood function (e.g., pseudo-likelihood) when the underlying model is non-Gaussian. Some of these works also provide finite sample bounds on the number of of samples required to correctly reconstruct the model structure. Unlike ours, these works focus on generative models, a difference

[8]Taken from Mulan (`http://mulan.sourceforge.net`)

that can be quite fundamental. For example, as we proved in Section 3, learning trees is a problem that is NP-hard in the M^3N setting while it is *polynomial* in the number of variables in the generative one. We note that we are not aware of finite sample results in the discriminative setting, where a different MRF is considered for each input x.

In the discriminative setting, Zhu et al. [2009] define an ℓ_1 regularized M^3N objective and present algorithms for its optimization. However, this approach does not consider the structure of the underlying graphical model over outputs y. Torralba et al. [2004] propose a greedy procedure based on boosting to learn the structure of a CRF, while Schmidt et al. [2008] present a block-ℓ_1 regularized pseudo-likelihood approach for the same task. While these methods do consider the structure of the graphical model, they do not attempt to produce tractable predictors. Further, they do not aim to learn models using a max-margin objective.

Bradley and Guestrin [2010] address the problem of learning trees in the context of conditional random fields. They show that using a particular type of tractable edge scores together with a maximum-spanning-tree (MST) algorithm may fail to find the optimal tree structure. Our work differs from theirs in several aspects. First, we consider the max-margin setting for structured prediction rather than the CRF setting. Second, they assume that the feature function $\phi(x, y)$ has a particular form where y and x are of the same order and where the outputs depend only on local inputs. Finally, we do not restrict our attention to MST algorithms. Consequently, our hardness result is more general and the approximations we propose are quite different from theirs. Finally, Chechetka and Guestrin [2010] also consider the problem of learning tree CRFs. However, in contrast to our work, they allow the structure of the tree to depend on the input x, which is somewhat more involved as it requires learning an additional mapping from inputs to trees.

7 Conclusion

We tackled the challenge of learning tree structured prediction models. To the best of our knowledge, ours is the first work that addresses the problem of structure learning in a discriminative M^3N setting. Moreover, unlike common structured sparsity approaches that are used in the setting of generative and conditional random field models, we explicitly target tree structures due to their appealing properties at test time. Following a proof that the task is NP-hard in general, we proposed a novel approximation scheme that relies on a circuit rank regularization objective that penalizes non-tree models, and that can be optimized using the CCCP algorithm. We demonstrated the effectiveness of our CRANK algorithm in several real-life domains. Specifically, we showed that CRANK obtained accuracies very close to those achieved by a full-dependence model, but with a much faster test time.

Many intriguing questions arise from our work: Under which conditions can we learn the optimal model, or guarantee approximation quality? Can we extend our framework to the case where the tree depends on the input? Can we use a similar approach to learn other graphs, such as low treewidth or high girth graphs? Such settings introduce novel forms of *graph-structured* sparsity which would be interesting to explore.

Acknowledgments

This research is funded by the ISF Centers of Excellence grant 1789/11, and by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI). Ofer Meshi is a recipient of the Google Europe Fellowship in Machine Learning, and this research is supported in part by this Google Fellowship.

A Realizing the parameters

Here we complete the hardness proof in Section 3 by describing the training set that realizes the parameters of Eq. (8) and Eq. (9). The approach below makes use of two training samples to constrain each parameter, one to upper bound it and one to lower bound it. Appealingly, the training samples are constructed in such a way that other samples do not constrain the parameter value, allowing us to realize the needed value for each and every parameter in the model.

The construction is similar to Sontag et al. [2010]. For all parameters it consists of the following steps: (i) define an assignment $x(y)$; (ii) identify two y values that potentially maximize the score; and (iii) show that these two complete assignments to x and y force the desired parameter value.

Preliminaries: Recall that the parameter vector θ is defined in Eq. (6) via a product of the features $\phi_i(y_i, x)$ and $\phi_{ij}(y_i, y_j, x)$ and the weights w_i and w_{ij} which do not depend on x or y and are shared across the parameters. For the hardness reduction, it will be convenient to set the features to indicator functions and show that the implied weight values are realizable. Specifically, we set the features to:

$$\begin{aligned}\phi_{ij}^{\text{off-diag}}(y_i, y_j) =& \mathbb{1}\{y_i \neq y_j\} \ \forall i, j\\ \phi_{ij}^{\text{diag}}(y_i, y_j) =& \mathbb{1}\{y_i = y_j = i \text{ or } y_i = y_j = j\} \ \forall i, j\\ \phi_1^{\text{bound}}(y_1) =& \mathbb{1}\{y_1 = 0\}\end{aligned}$$

Recall that to realize the desired parameters θ, we need to introduce training samples such that for all i, j:

$$w_{ij}^{\text{off-diag}} = -n^2, \quad w_{ij}^{\text{diag}} = 1, \quad w_1^{\text{bound}} = D,$$

with the dimension of w equalling $2|E(G)| + 1$.

Finally, using N_i to denote the set of neighbors of node i in the graph G, we will use the following (large) value to force variables not to take some chosen states:

$$\gamma_i = \begin{cases} 1 + |N_i|(n^2+1) & i \neq 1 \\ 1 + |N_i|(n^2+1) + D & i = 1 \end{cases}$$

Realizing the weight w_1^{bound}: We define $x(y)$ as:

$$x_1(y_1) = \begin{cases} 0 & y_1 = 0 \\ -D & y_1 = 1 \\ -\gamma_1 & y_1 \geq 2 \end{cases}$$

$$x_i(y_i) = \begin{cases} 0 & y_i = 2 \\ -\gamma_i & y_i \neq 2 \end{cases} \qquad \text{for all } i \neq 1$$

Recalling the definition of γ above, this implies that the only assignments to y that can maximize the score of Eq. (6) are $(0, 2, 2, ..., 2)$ and $(1, 2, 2, ..., 2)$. In particular, we have:

$$S(0,2,2,...,2; x, w) = \sum_{k \in N_1} w_{1k}^{\text{off-diag}} + x_1(0) + \bar{S}$$
$$S(1,2,2,...,2; x, w) = w_1^{\text{bound}} + \sum_{k \in N_1} w_{1k}^{\text{off-diag}} + x_1(1) + \bar{S}$$

where $\bar{S}$ is the sum of all components that do not involve the first variable.

For the final step we recall that the weights w have to satisfy the constraints: $S(y^m; x^m, w) \geq S(y; x^m, w)$ for all m, y. Thus, we will define two instances (x^m, y^m) for which some y assignment will constrain the weight as needed (in both cases, x^m is defined as above). When $y^{(m)} = (0, 2, 2, \ldots, 2)$, the assignment $y = (1, 2, 2, \ldots, 2)$ yields $w_1^{\text{bound}} \leq D$ and all other assignments do no further constrain the weight. Similarly, for $y^{(m')} = (1, 2, 2, \ldots, 2)$, the assignment $y = (0, 2, 2, \ldots, 2)$ yields $w_1^{\text{bound}} \geq D$. Together, the two assignments constrain the weight parameter to $w_1^{\text{bound}} = D$, as desired.

Realizing the weights $w_{ij}^{\text{off-diag}}$: We define $x(y)$ as:

$$x_i(y_i) = \begin{cases} 0 & y_i = 0 \\ -\gamma_i & y_i \neq 0 \end{cases}, \quad x_j(y_j) = \begin{cases} 0 & y_j = 0 \\ n^2 & y_j = 1 \\ -\gamma_j & y_j \geq 2 \end{cases}$$

$$x_k(y_k) = \begin{cases} 0 & y_k = k \\ -\gamma_k & y_k \neq k \end{cases} \qquad \text{for all } k \neq i, j$$

This implies that except for i and j, all y_k's must take their corresponding assignment so that $y_k = k$. W.l.g., suppose that $i = 1$ and $j = 2$. The only assignments that can maximize the score are $(0, 0, 3, 4, 5, ..., n)$ and $(0, 1, 3, 4, 5, ..., n)$ with values:

$$S(0,0,3,4,5,...,n; x, w) =$$
$$\sum_{k \in N_i} w_{ik}^{\text{off-diag}} + \sum_{k' \in N_j} w_{jk'}^{\text{off-diag}} + x_i(0) + x_j(0) + \bar{S}$$
$$S(0,1,3,4,5,...,n; x, w) =$$
$$w_{ij}^{\text{off-diag}} + \sum_{k \in N_i} w_{ik}^{\text{off-diag}} + \sum_{k' \in N_j} w_{jk'}^{\text{off-diag}} + x_i(0) + x_j(1) + \bar{S}$$

As before, setting $y^{(m)} = (0, 0, 3, 4, 5, ..., n)$ and then $y^{(m')} = (0, 1, 3, 4, 5, ..., n)$ yields the constraint $w_{ij}^{\text{off-diag}} = -n^2$.

Realizing the weights w_{ij}^{diag}: We define $x(y)$ as:

$$x_i(y_i) = \begin{cases} 0 & y_i = i \\ -\gamma_i & y_i \neq i \end{cases}, \quad x_j(y_j) = \begin{cases} n^2 & y_j = 0 \\ -1 & y_j = i \\ -\gamma_j & y_j \notin \{0, i\} \end{cases}$$

$$x_k(y_k) = \begin{cases} 0 & y_k = k \\ -\gamma_k & y_k \neq k \end{cases} \qquad \text{for all } k \neq i, j$$

As before, for all $k \neq i, j$ the assignment is forced to $y_k = k$. The maximizing assignments are now $(1, 0, 3, 4, 5, ..., n)$ and $(1, 1, 3, 4, 5, ..., n)$ (assuming w.l.g., $i = 1, j = 2$) with score values:

$$S(1,0,3,4,5,...,n; x, w) =$$
$$w_{ij}^{\text{off-diag}} + \sum_{k \in N_i} w_{ik}^{\text{off-diag}} + \sum_{l \in N_j} w_{jl}^{\text{off-diag}} + x_i(i) + x_j(0) + \bar{S}$$
$$S(1,1,3,4,5,...,n; x, w) =$$
$$w_{ij}^{\text{diag}} + \sum_{k \in N_i} w_{ik}^{\text{off-diag}} + \sum_{l \in N_j} w_{jl}^{\text{off-diag}} + x_i(i) + x_j(i) + \bar{S}$$

Now, setting $y^{(m)} = (1, 0, 3, 4, 5, ..., n)$, the assignment $y = (1, 1, 3, 4, 5, ..., n)$ implies $w_{ij}^{\text{off-diag}} + n^2 \geq w_{ij}^{\text{diag}} - 1$. Since we already have that $w_{ij}^{\text{off-diag}} = -n^2$, we obtain $w_{ij}^{\text{diag}} \leq 1$. Similarly, adding $y^{(m')} = (1, 1, 3, 4, 5, ..., n)$ implies $w_{ij}^{\text{diag}} \geq 1$, so together we have $w_{ij}^{\text{diag}} = 1$, as required.

Importantly, our trainset realizes the same edge parameters for any possible tree, since edge weights are constrained independently of other edges.

References

H. Aissi, C. Bazgan, and D. Vanderpooten. Approximation complexity of min-max (regret) versions of shortest path, spanning tree, and knapsack. In *Proceedings of the 13th annual European conference on Algorithms*, pages 862–873, Berlin, Heidelberg, 2005. Springer-Verlag.

F. Bach. Structured sparsity-inducing norms through submodular functions. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 118–126. 2010.

G. H. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data*. The MIT Press, 2007.

C. Berge. *The Theory of Graphs*. Dover Books on Mathematics Series. Dover, 1962.

J. K. Bradley and C. Guestrin. Learning tree conditional random fields. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 127–134. Omnipress, 2010.

A. Chechetka and C. Guestrin. Evidence-specific structures for rich tractable crfs. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 352–360. 2010.

C. I. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.

M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, 2002.

K. Dembczynski, W. Cheng, and E. Hüllermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In *ICML*, pages 279–286, 2010.

T. Finley and T. Joachims. Training structural SVMs when exact inference is intractable. In *Proceedings of the 25th International Conference on Machine learning*, pages 304–311, 2008.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

E. Hazan and S. Kale. Beyond convexity: Online submodular minimization. In *Advances in Neural Information Processing Systems 22*, pages 700–708. 2009.

J. B. Kruskal. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Math. Society*, 7(1):48–50, 1956.

S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *Proceedings of The 30th International Conference on Machine Learning*, pages 53–61, 2013.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th Int. Conf. on Machine Learning*, pages 282–289, 2001.

S.-I. Lee, V. Ganapathi, and D. Koller. Efficient structure learning of Markov networks using L1-regularization. In *Advances in Neural Information Processing Systems (NIPS 2006)*, 2007.

O. Meshi, D. Sontag, T. Jaakkola, and A. Globerson. Learning efficiently with approximate inference via dual losses. In *ICML*, pages 783–790, New York, NY, USA, 2010. ACM.

J. G. Oxley. *Matroid Theory*. Oxford University Press, 2006.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

P. Ravikumar, G. Raskutti, M. J. Wainwright, and B. Yu. Model selection in gaussian graphical models: High-dimensional consistency of l1-regularized MLE. In *Advances in Neural Info. Processing Systems 17*. 2008.

P. Ravikumar, M. J. Wainwright, and J. Lafferty. High-dimensional ising model selection using l1-regularized logistic regression. *Annals of Statistics*, 38(3):1287–1319, 2010.

M. Schmidt, K. Murphy, G. Fung, and R. Rosales. Structure learning in random fields for heart motion abnormality detection. In *CVPR*, pages 1 –8, 2008.

Y. Shimony. Finding the MAPs for belief networks is NP-hard. *Aritifical Intelligence*, 68(2):399–410, 1994.

D. Sontag, O. Meshi, T. Jaakkola, and A. Globerson. More data means less inference: A pseudo-max approach to structured learning. In *Advances in Neural Information Processing Systems 23*, pages 2181–2189. 2010.

P. D. Tao and L. T. H. An. Convex analysis approach to dc programming: theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems*. MIT Press, 2003.

A. Torralba, K. P. Murphy, and W. T. Freeman. Contextual models for object detection using boosted random fields. In *Advances in Neural Information Processing Systems 17*, pages 1401–1408. 2004.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6(2):1453, 2006.

M. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers Inc., Hanover, MA, USA, 2008.

A. L. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Comput.*, 15(4):915–936, Apr. 2003.

Y. Zhang and J. Schneider. Maximum margin output coding. In *ICML*, 2012.

J. Zhu, E. P. Xing, and B. Zhang. Primal sparse max-margin markov networks. In *Proceedings of the ACM SIGKDD international conf. on Knowledge discovery and data mining*, KDD '09, pages 1047–1056, 2009.

Tighter Linear Program Relaxations for High Order Graphical Models

Elad Mezuman[*] Hebrew University Jerusalem, Israel

Daniel Tarlow[*] Microsoft Research Cambridge, UK

Amir Globerson Hebrew University Jerusalem, Israel

Yair Weiss Hebrew University Jerusalem, Israel

Abstract

Graphical models with High Order Potentials (HOPs) have received considerable interest in recent years. While there are a variety of approaches to inference in these models, nearly all of them amount to solving a linear program (LP) relaxation with *unary consistency* constraints between the HOP and the individual variables. In many cases, the resulting relaxations are loose, and in these cases the results of inference can be poor. It is thus desirable to look for more accurate ways of performing inference. In this work, we study the LP relaxations that result from enforcing additional consistency constraints between the HOP and the rest of the model. We address theoretical questions about the strength of the resulting relaxations compared to the relaxations that arise in standard approaches, and we develop practical and efficient message passing algorithms for optimizing the LPs. Empirically, we show that the LPs with additional consistency constraints lead to more accurate inference on some challenging problems that include a combination of low order and high order terms.

1 Introduction

Graphical models are an excellent tool for expressing models that arise in a wide variety of domains including computational biology, natural language processing, and computer vision. A long-standing research challenge is to expand the range of problems that can be expressed with graphical models such that learning and inference can be performed efficiently. Recently, there has been a resurgence of interest in

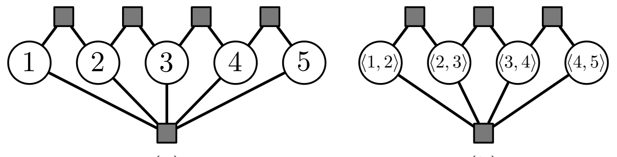


Figure 1.1: We show that LP-based message passing in graphical models with high order potentials (HOPs) is likely to be poor when the HOP communicates via a single-variable interface as in (a). Communicating with a subset of edges of the same model leads to much better results. For example, in (b), it is provably exact.

high order potentials (HOPs), which generally refer to modeling components that have tractable structure that is not revealed by looking at the graphical structure of the model. These approaches are rooted in earlier work on graphical models like Pearl's polytree algorithm (Pearl, 1988), noisy-OR interactions (Heckerman, 1989), and context specific independencies (Boutilier et al., 1996). Their recent popularity has been fueled by the availability of efficient routines for using HOPs within modern linear program (LP)-based message passing algorithms such as dual decomposition (Komodakis et al., 2007), MPLP (Globerson and Jaakkola, 2007), and convex belief propagation (Weiss et al., 2007). Recent works such as (Tarlow et al., 2010) attempt to categorize general classes of HOP and give efficient algorithms for using them within message passing algorithms. There are many specific applications, such as using HOPs in an image segmentation task to jointly optimize over the appearance model and segmentation (Vicente et al., 2009).

Despite the success that has been achieved with message passing algorithms and HOPs, nearly all of the approaches are equivalent to a particular LP relaxation (Koller and Friedman, 2009). In addition, several other approaches to dealing with HOPs, such as reducing them to low order models, equate to the same relaxation. We expand on these equivalences in Sec. 4.

[*]Equal contribution

The main goal of this paper is to show that the LP relaxation resulting from the standard approach is weak, and to propose an alternative that maintains many of the same desirable computational efficiencies while leading to more accurate inference.

The paper proceeds as follows. We begin by establishing the ubiquity of the relaxation that we term the *unary consistency LP*, showing that many approaches to dealing with HOPs are equivalent to this relaxation. Having established this, we go on to show that the unary consistency relaxation is quite weak. We provide several examples and some analysis to help understand when this is the case and why it fails. Next, we introduce a family of LPs that provide tighter relaxations than the standard relaxation, but (as we show) still admit efficient algorithms for optimizing them. We provide theoretical analysis of these LPs, showing when they are provably tight.

We then turn to practical concerns and show that our tighter LPs can often be efficiently solved using standard message-passing algorithms: the main difference is that the nodes corresponding to the higher order potential now receive and send messages that are functions of pairs of variables, not just singletons. This makes the message-passing more involved but we identify special cases of HOPs for which the message computation is still tractable. We also show how to use the messages to compute tighter bounds on the MAP and how to choose which pairs of variables should be added in a way that is guaranteed to tighten the bound. We illustrate the performance of our method on both synthetic models and real image segmentation problems.

2 Motivating Examples

To begin, we will establish notation, then consider two concrete examples that illustrate the looseness of the standard LP relaxation for dealing with HOPs.

Notation and Preliminaries We let an energy function over n discrete variables $\mathbf{x} = \{x_1, ..., x_n\}$ be defined as follows. Given a graph $G = (\mathcal{V}, \mathcal{E})$ with n vertices, there are potentials $\theta_i(x_i), \theta_{ij}(x_i, x_j)$ for each vertex and each edge in the graph, respectively, and one HOP over all the variables, $\theta_\alpha(\mathbf{x})$. We wish to find the minimum energy configuration (alternatively the *maximum a posteriori* (MAP) assignment):

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{ij \in \mathcal{E}} \theta_{ij}(x_i, x_j) + \theta_\alpha(\mathbf{x}). \quad (2.1)$$

This is an integer program that is NP hard to solve in general. The LP relaxation approach works in two stages. First, the integer program is converted into a linear program with the same solution, but which requires exponentially many constraints to express the domain. Then the linear program is relaxed by outer bounding the domain to yield the $\mathbf{LP}_\emptyset$ below. For more background on LP relaxations, we recommend Wainwright and Jordan (2008), Koller and Friedman (2009) and Sontag et al. (2010).

We now review the standard LP relaxation approach to approximating Eq. 2.1. The relaxation maintains three types of distributions: over single variables, over pairs of variables, and over all of x. All three are constrained to agree on their singleton marginals. The singleton, pairwise and HOP terms are then replaced by their expectation according to the corresponding distributions. We call this approach the *Unary Consistency LP* and denote it by $\mathbf{LP}_\emptyset$. The resulting optimization problem is

$\mathbf{LP}_\emptyset$ (Unary Consistency LP)

$$\min_q \sum_{i \in \mathcal{V}} \mathbb{E}_{q_i}[\theta(x_i)] + \sum_{ij \in \mathcal{E}} \mathbb{E}_{q_{ij}}[\theta_{ij}(x_i, x_j)] + \mathbb{E}_{q_\alpha}[\theta_\alpha(\mathbf{x})]$$
$$\text{s.t.} \ \sum_{x_i} q_{ij}(x_i, x_j) = q_j(x_j) \ , \ \sum_{x_j} q_{ij}(x_i, x_j) = q_i(x_i)$$
$$\sum_{\mathbf{x}: \mathbf{x}(i) = x_i} q_\alpha(\mathbf{x}) = q_i(x_i) \qquad \forall i \in \mathcal{V}, \quad (2.2)$$

where we omit for space (as we will throughout) the additional constraints that q variables are non-negative and sum to 1, and some of the quantifications (e.g., $ij \in \mathcal{E}$ and the values of x_i in the last constraint). As we show in Sec. 4, this relaxation is commonly used because it can be solved efficiently for several families of HOPs.

Introductory Examples We start with two simple instances of MAP problems with tractable HOP for which the standard approach fails. In both cases the factor graph (Fig. 1.1a) consists of a simple chain and a cardinality-based potential which is amenable to the techniques described in (Tarlow et al., 2010) for solving $\mathbf{LP}_\emptyset$. Unfortunately, the relaxation is loose, which is manifested in a fractional solution that has better objective than the original integer program solution. This means the relaxation is inaccurate, and, more importantly, it prevents us from finding a good solution to the original problem.

In both examples $x_i \in \{0, 1\}$ and the model is a chain with an even number of nodes and attractive pairwise potentials of the form $\theta_{i,i+1} = \begin{pmatrix} 0 & c \\ c & 0 \end{pmatrix}$ (where $c > 0$).

1. Finding the 2nd best assignment (Fromer and Globerson, 2009) using an exclusion factor. If we add a local potential that favors the binary variables being off: $\theta_i = \begin{pmatrix} 0 & \epsilon \end{pmatrix}^\top$ with $\epsilon > 0$, then

clearly the best assignment is $\mathbf{x} = \overrightarrow{0}$. We add an exclusion potential to exclude this assignment: $\theta_\alpha(\mathbf{x}) = \infty$ when $\mathbf{x} = \overrightarrow{0}$ and 0 otherwise. and now the MAP of the problem with the high-order potential is the second best assignment in the original problem. Assuming $c \gg \epsilon$, the 2^{nd} best is $\mathbf{x} = \overrightarrow{1}$ with an energy of $n\epsilon$. However, the $\mathbf{LP}_\emptyset$ solution has a value of ϵ and corresponds to the following fractional optimum: $q_i = \begin{pmatrix} \frac{n-1}{n} & \frac{1}{n} \end{pmatrix}^\top$, $q_{i,i+1} = \begin{pmatrix} \frac{n-1}{n} & 0 \\ 0 & \frac{1}{n} \end{pmatrix}$, and $q_\alpha(\mathbf{x}) = \frac{1}{n}$ for all the assignments in which exactly one variable is on.

2. Partitioning graphs using average-cut (Mezuman and Weiss, 2012). Here the HOP prefers assignments with similar number of off and on variables: $\theta_\alpha(\mathbf{x}) = -\lambda \cdot |\mathbf{x}| \cdot (n - |\mathbf{x}|)$. The optimal integral solution is to break the chain in the middle with value of $c - \lambda(\frac{n}{2})^2$, but the $\mathbf{LP}_\emptyset$ solution has a value of $-\lambda(\frac{n}{2})^2$ and is again fractional: $q_i = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$, $q_{i,i+1} = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix}$, and $q_\alpha(\mathbf{x}) = \frac{1}{M}$ for all $M = \binom{n}{n/2}$ assignments that have $n/2$ ones.

Notice that the LP solution is not only fractional, but is in fact also completely "tied": its solution gives us no hint as to the true MAP.

3 Related work

Applications of High Order Potentials HOPs can be used to incorporate nonlocal structure into a model. In recent years, there have been many works that incorporate these types of interactions. They are particularly useful for modelling highly structured global interactions like those that arise in models for parsing sentences (Smith and Eisner, 2008; Koo et al., 2010; Martins et al., 2010), in models for image segmentation to enforce connectivity constraints (Nowozin and Lampert, 2009) or higher order smoothness (Kohli et al., 2007; Gould, 2011), and in models of textures to encourage soft pattern matching (Rother et al., 2009). They arise when "collapsing" certain models, like in (Vicente et al., 2009), where optimizing out an image segmentation appearance model leads to an energy function over segmentations that has high order terms. They have also been used to solve balanced graph partitioning problems (Mezuman and Weiss, 2012) and to enforce constraints over latent variable activations in e.g., Restricted Boltzmann Machines (Swersky et al., 2012).

Tighter Linear Program Relaxations The canonical works on tightening LP relaxations using message-passing come from Sontag et al. (2008); Werner (2008) and Komodakis and Paragios (2008), and were followed up in several works, such as Batra et al. (2011); Sontag et al. (2012). As discussed in Sontag (2010), at their core, these approaches can be viewed as searching in similar ways for additional consistency constraints to enforce such that adding them to the LP leads to a tighter relaxation. While the general approach is applicable in the LP relaxations with HOPs that we consider here, there are computational challenges that must be addressed in order to do this search efficiently. We develop the needed methods in this work. We note that while Werner (2008) discusses HOPs in the context of the max-sum diffusion algorithm and the class of LPs that we study here can be expressed within the framework presented there, the final suggestion for working with HOPs is to use the unary consistency LP, and a proof is provided that if the model is submodular and the HOP is submodular, then the LP is tight.

Fromer and Globerson (2009) deal with the case of excluding a single joint assignment. We will show that the baseline they consider, Santos, is equivalent to unary consistency, and that their method (which leads to tight relaxations on trees) is a special case of our method. Thus, we can get an equally tight relaxation by using the approach proposed in this paper. Another special case of note where tighter HOP relaxations have been discussed is Komodakis and Paragios (2009). There, they employ a merging strategy for dealing with several pattern-based HOPs, in that they show that patterns along rows and columns of a grid can be combined into a single HOP where messages can still be computed efficiently. However, after this merging, only unary consistency is enforced. For the purposes of this paper, we assume throughout that if any tractability-preserving merging of HOPs is possible, it has already been done.

4 Unary Consistency Linear Programs

Many existing methods for inference with HOPs are equivalent to $\mathbf{LP}_\emptyset$. We highlight some of these below.

Message Passing with the standard Factor Graph Perhaps the most common method for solving MAP inference in graphical models with HOPs is to build the factor graph and pass messages between variables and factors. For certain higher-order potentials, the messages between the variable nodes and the HOP node can be calculated efficiently (Tarlow et al., 2010).

Different works use somewhat different update schemes. One option is to use loopy belief propagation (Yedidia et al., 2005). However, there are typically no performance guarantees in this case (e.g., no convergence results or optimality certificates). A different class of structurally similar algorithms retain the message-passing flavor of BP while also giving an optimality certificate (Globerson and Jaakkola, 2007; Werner, 2007; Kolmogorov, 2006; Komodakis et al., 2010; Sontag et al., 2010; Weiss et al., 2007). These can all be shown to be solving $\mathbf{LP}_\emptyset$.

Simplification with Auxiliary Variables An alternative strategy for dealing with certain HOPs is to create auxiliary variables in such a way as to reduce the problem to a pairwise problem, and then solve the pairwise problem using the standard pairwise LP relaxation. Here, we study the strength of the LP relaxations that result from this strategy. For example, Kohli et al. (2009), Gould (2011), and Rother et al. (2009) all follow this or a closely related approach.

The approach proceeds as follows. Start with a HOP and some unary and pairwise potentials:

$$E(\mathbf{x}) = \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) + \theta_\alpha(\mathbf{x}). \qquad (4.1)$$

Next, introduce an auxiliary variable z such that minimizing it out leaves the energy over $\mathbf{x}$ unchanged. Namely, $\min_z \theta_{z,\alpha}(z, \mathbf{x}) = \theta_\alpha(\mathbf{x})$. We then have:

$$E(\mathbf{x}) = \min_z \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) + \theta_{z,\alpha}(z, \mathbf{x}).$$

Finally, it often holds that given z, the HOP becomes fully factorized, i.e., $\min_z \theta_{z,\alpha}(z, \mathbf{x}) = \min_z \sum_i \theta_{zi}(z, x_i)$, so that $E(\mathbf{x})$ is given by:

$$\min_z \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) + \sum_i \theta_{zi}(z, x_i). \qquad (4.2)$$

At this point, the minimization over x can be done jointly with the minimization over z using the following LP relaxation, which we call $\mathbf{LP_{red}}$:

$\mathbf{LP_{red}}$

$$\min_q \sum_{i\in\mathcal{V}} (\mathbb{E}_{q_i}[\theta(x_i)] + \mathbb{E}_{q_{zi}}[\theta_{zi}(z, x_i)]) + \sum_{ij\in\mathcal{E}} \mathbb{E}_{q_{ij}}[\theta_{ij}(x_i, x_j)]$$

$$\sum_{x_j} q_{ij}(x_i, x_j) = q_i(x_i) \quad \sum_{x_i} q_{ij}(x_i, x_j) = q_j(x_j) \quad (4.3)$$

$$\sum_z q_{zi}(z, x_i) = q_i(x_i) \quad \sum_{x_i} q_{zi}(z, x_i) = q_z(z) \qquad (4.4)$$

This relaxation seems quite different from $\mathbf{LP}_\emptyset$. However, we have the surprising result that they are in fact equivalent (the proof is in the appendix).

Proposition 1. *The relaxations* $\mathbf{LP}_\emptyset$ *and* $\mathbf{LP_{red}}$ *are equivalent. Namely, they have the same objective value, and there is a mapping between their optima.*

In the case that more than one auxiliary variable is created by the pairwise transformation, the unary consistency LP will be at least as tight as the reduced LP, but they are no longer equal in general. A corollary of this analysis is that if the pairwise transformation introduces only submodular pairwise terms (and the pairwise part of the model is submodular), then the unary consistency LP is tight. This is closely related to (but less general than) the result proved in (Werner, 2008).

Exclusion Potentials and the Santos Inequality A special case of HOP model that has received significant attention is where a model is modified so as to exclude a single joint assignment $\mathbf{x}^*$ (e.g., the first introductory example). In this context, several LP relaxations have been proposed (e.g., Fromer and Globerson, 2009). It is thus interesting to ask which of these is equivalent to $\mathbf{LP}_\emptyset$. It turns out that $\mathbf{LP}_\emptyset$ corresponds to an LP with no $q_\alpha(\mathbf{x})$ variables, but rather a single constraint (in addition to the pairwise consistency constraints): $\sum_i q_i(x_i^*) \le n - 1$, which was first suggested in Santos Jr (1991). Intuitively, it states that at most $n - 1$ of the variables can agree with the assignment $\mathbf{x}^*$. The proof of the equivalence to $\mathbf{LP}_\emptyset$ is straightforward and follows from the characterization of the assignment excluding polytope for an empty graph, and its relation to the Santos inequality (see Fromer and Globerson, 2009).

5 Tighter Linear Programs

In this section, we introduce the family of tighter LP relaxations that are the focus of this work, and we study their theoretical properties. We begin by defining a family of LPs that are tighter relaxations than $\mathbf{LP}_\emptyset$, and then we will prove a tightness result.

Let $S \subseteq \mathcal{E}$ be a subset of the edges in G, and define $\mathcal{V}(S)$ to be the set of variables that appear in at least one edge in S. We can then define an LP that enforces consistency between the HOP and the edges in S, while maintaining unary consistency with variables in $\mathcal{V} - \mathcal{V}(S)$:

$\mathbf{LP}_S$ (Partial Edge Consistency LP)

$$\min_q \sum_{i\in\mathcal{V}} \mathbb{E}_{q_i}[\theta(x_i)] + \sum_{ij\in\mathcal{E}} \mathbb{E}_{q_{ij}}[\theta_{ij}(x_i, x_j)] + \mathbb{E}_{q_\alpha}[\theta_\alpha(\mathbf{x})]$$

$$\text{s.t.} \ \sum_{x_i} q_{ij}(x_i, x_j) = q_j(x_j) \ , \ \sum_{x_j} q_{ij}(x_i, x_j) = q_i(x_i)$$

$$\sum_{\mathbf{x}:\mathbf{x}(i)=x_i} q_\alpha(\mathbf{x}) = q_i(x_i) \qquad \forall i \in \mathcal{V} - \mathcal{V}(S)$$

$$\sum_{\mathbf{x}:\mathbf{x}(i)=x_i,\mathbf{x}(j)=x_j} q_\alpha(\mathbf{x}) = q_{ij}(x_i, x_j) \qquad \forall ij \in S$$

At one extreme, when $S = \emptyset$, $\mathbf{LP}_S$ is equal to $\mathbf{LP}_\emptyset$. At the other extreme, consistency is enforced between the HOP and all edges, yielding the following simplified LP, which will be of special interest:

$\mathbf{LP}_\mathcal{E}$ (Full Edge Consistency LP)

$$\begin{aligned}\min_q &\sum_{i\in\mathcal{V}} \mathbb{E}_{q_i}[\theta(x_i)] + \sum_{ij\in\mathcal{E}} \mathbb{E}_{q_{ij}}[\theta_{ij}(x_i,x_j)] + \mathbb{E}_{q_\alpha}[\theta_\alpha(\mathbf{x})] \\ \text{s.t. } &\sum_{x_i} q_{ij}(x_i,x_j) = q_j(x_j)\ ,\ \sum_{x_j} q_{ij}(x_i,x_j) = q_i(x_i) \\ &\sum_{\mathbf{x}:\mathbf{x}(i)=x_i,\mathbf{x}(j)=x_j} q_\alpha(\mathbf{x}) = q_{ij}(x_i,x_j) \qquad \forall ij\in\mathcal{E}\end{aligned}$$

5.1 Strength of $\mathbf{LP}_\mathcal{E}$

We begin with the simple observation that $\mathbf{LP}_\mathcal{E}$ is always tight.

Proposition 2. *$\mathbf{LP}_\mathcal{E}$ is tight.*

Proof. Since with all the pairwise constraints the expectation of θ_i and θ_{ij} under q_α is the same as under q_i and q_{ij}, respectively, $\mathbf{LP}_\mathcal{E}$ is equivalent to

$$\min_q \mathbb{E}_{q_\alpha}[\theta_\alpha(\mathbf{x}) + \theta(x_i) + \theta_{ij}(x_i,x_j)]. \tag{5.1}$$

This LP has an integer solution (i.e., it is tight) because it is always better to put all the q_α mass on the best assignment than to divide it. $\square$

Thus, the space of LP relaxations that can be constructed as $\mathbf{LP}_S$ for some choice of edge set S range from the standard but weak $\mathbf{LP}_\emptyset$ to the tight $\mathbf{LP}_\mathcal{E}$. This justifies our focus on this family of $\mathbf{LP}_S$.

6 Optimization with Message Passing

As we reviewed earlier, $\mathbf{LP}_\emptyset$ can be solved by a variety of message-passing algorithms operating on the standard factor graph, where factor nodes communicate with individual variable nodes. Similarly, it is easy to show that the same algorithms can solve $\mathbf{LP}_S$ when they are applied on a modified factor graph where the node corresponding to the HOP communicates with *pairs of nodes* which correspond to edges in S (see Fig. 1.1b). The key question is the complexity of calculating the messages to and from the HOP. We now identify cases where these messages can be computed efficiently.

We begin by recalling the dual of $\mathbf{LP}_S$. The dual variables are $\delta_{ij}(x_i,x_j)$ for $ij\in S$ (which we interpret as messages between the factor α and the edges in S) and $\delta_i(x_i)$ for $i\in\mathcal{V}$ (messages between the factor α and singletons). The dual problem is to maximize $B(\delta)$, a lower bound on the MAP:

$$B(\delta) = \sum_i \min_{x_i} \tilde{\theta}_i^\delta(x_i) + \sum_{ij} \min_{x_i,x_j} \tilde{\theta}_{ij}^\delta(x_i,x_j) + \min_x \tilde{\theta}_\alpha^\delta(\mathbf{x}), \tag{6.1}$$

where $\tilde{\theta}^\delta$ is a reparameterization of the original energy function:

$$\tilde{\theta}_\alpha^\delta(\mathbf{x}) = \theta_\alpha(\mathbf{x}) - \sum_{ij\in S} \delta_{ij}(x_i,x_j) - \sum_{i\in\mathcal{V}-\mathcal{V}(S)} \delta_i(x_i).$$

Expressions for $\tilde{\theta}_i^\delta(x_i), \tilde{\theta}_{ij}^\delta(x_i,x_j)$ are similar (Sontag et al., 2010). Most message passing approaches for solving this problem iteratively update δ to increase the bound. All these message update schemes require solving $\min_x \tilde{\theta}_\alpha^\delta(\mathbf{x})$ for arbitrary values of δ, or calculating its min-marginals (see Sontag et al., 2010, for a thorough review of such approaches). For general $\theta_\alpha(\mathbf{x})$ this is of course difficult. Below we highlight some cases where it is tractable, and therefore $\mathbf{LP}_S$ can be solved efficiently with message passing.

- Low tree-width S graphs with cardinality-based potentials. When $\theta_\alpha(\mathbf{x})$ is a cardinality potential (i.e., $\theta_\alpha(\mathbf{x}) = f(\sum_i x_i)$, where $f(\cdot)$ is some arbitrary function) and S is a tree-structured subset of edges, then this is closely related to one of the problems considered by (Tarlow et al., 2012). There, it was shown that messages to and from the HOP can be calculated by performing exact inference on an augmented tree graph with complexity that is at most $O(n^2)$. This result is easily extended to the case where S forms a low tree-width graph and the messages can be computed exactly in a time that is exponential in the tree-width of S.

- Low tree-width S graphs with Pattern HOPs. Another HOP that has received interest are the pattern potentials of (Rother et al., 2009). Here, the potentials are of the form $\theta_\alpha(\mathbf{x}) = \min_{k\in\{1,\ldots,K\}} \sum_i w_i^{(k)} x_i$ where each real-valued vector $w^{(k)}$ can be thought of as encoding a pattern that is desirable to match. This potential is actually quite simple to work with, by noting that $\min_x \tilde{\theta}_\alpha^\delta(\mathbf{x})$ is equivalent to:

$$\min_{k,x}\left[\sum_i w_i^{(k)} x_i - \sum_{ij\in S} \delta_{ij}(x_i,x_j) - \sum_{i\in\mathcal{V}-\mathcal{V}(S)} \delta_i(x_i)\right].$$

  From here, it is clear that the argmin or the min-marginals can be computed by constructing a junction tree over S, then solving K different problems where for problem k, the unary potentials have been modified by $w^{(k)}$, then taking the elementwise minima.

A simple corollary of the above discussion is that whenever the graphical model has low tree-width and the HOP is either cardinality-based or a pattern HOP then $\mathbf{LP}_{\mathcal{E}}$ can be efficiently computed. In particular, for the two motivating examples discussed in the introduction, running message-passing on the factor graph shown in Fig. 1.1b can be performed efficiently and is guaranteed to provide the MAP.

7 Choosing a Tractable Edge Set

When the graph has high tree-width, we cannot efficiently solve $\mathbf{LP}_{\mathcal{E}}$, and a natural question is how to choose a subset of edges S such that $\mathbf{LP}_S$ is as tight as possible, but can still be solved in practice. For the HOPs we consider, this will be the case as long as S has low tree-width (see Sec. 6).

As mentioned earlier, this problem has been studied generally in several works, including Sontag et al. (2008), Werner (2008) and Komodakis and Paragios (2009). When trying to adapt these approaches to models with HOPs that are based on $\mathbf{LP}_S$, the general methodology stays the same, but as with the message updates, computational challenges arise. Specifically, the above methods are all based around finding additional consistency constraints to add that are guaranteed to improve $B(\delta)$. In our context, the motivation is the following Lemma:

Lemma 1. *Suppose we have solved the dual of* $\mathbf{LP}_S$ *with some set of edges S. If there exists an edge $(i,j) \notin S$ for which there is no overlap between the minimizing assignments of $\tilde{\theta}_{ij}$ and the minimizing assignments of $\tilde{\theta}_\alpha$, then defining $T = \{S \cup (i,j)\}$ we have* $\mathbf{LP}_S < \mathbf{LP}_T$ *(i.e. adding that edge to S will lead to a strictly tighter LP relaxation).*

The proof follows from using the reparameterization given by the dual variables of $\mathbf{LP}_S$ to construct a valid reparameterization for the dual of $\mathbf{LP}_T$ and the dual value will be strictly higher. Existing methods are often based on similar reasoning (e.g., this is essentially the same result that appears in Werner (2008) in the discussion of cutting planes).

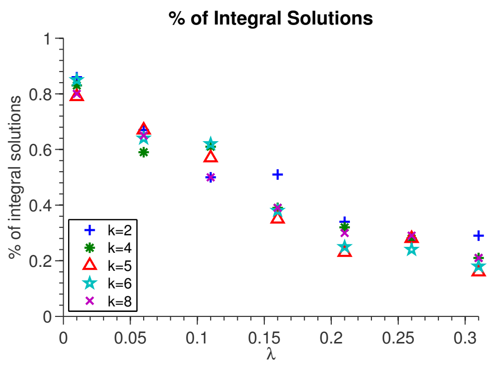
Sequentially Adding Edges using Weak Cycle Agreement (WCA) Recall that our goal is to find a low tree-width S such that $\mathbf{LP}_S$ is as tight as possible. Motivated by Lem. 1 we use the following procedure for approximating such a set. Start with an edge set, S, with tree-width one. Then, keep adding edges as long as the tree-width stays small. The edges added are those that satisfy the condition in Lem. 1 and hence result in strict increase of the LP objective. In what follows we provide additional details.

To obtain the initial tree we follow a simple heuristic. Calculate weights $w_{ij} = \max(\theta_{ij}) - \min(\theta_{ij})$ for each edge, and find the spanning tree S with the maximum overall weight. The rationale is that edges with close to uniform potentials (i.e., low w_{ij}) are more likely to be consistent with the HOP.

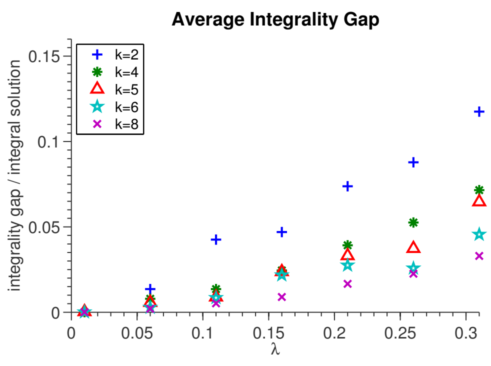
Next, we add K edges in each iteration using the following procedure. Run $\mathbf{LP}_S$ to convergence, and find the *set* of M assignments that minimize the HOP term of the reparameterization: $\{\mathbf{x}^m\}_{m=1}^M = \arg\min_{\mathbf{x}} \tilde{\theta}_\alpha(\mathbf{x})$. As long as M is small—as we found it is in practice—this can be done efficiently by following all back-pointers when decoding from the junction tree structures used for computing message updates. Next, for each edge $ij \in \mathcal{E} \setminus S$ whose addition to S does not violate the maximum tree-width, we compute its *weak cycle agreement* (WCA) measure: $\min_{\mathbf{x}^m} \tilde{\theta}_{ij}(x_i^m, x_j^m) - \min_{x_i,x_j} \tilde{\theta}_{ij}(x_i, x_j)$. By Lem. 1, addition of any edge with WCA > 0 will give a tighter relaxation. If many edges have WCA > 0, we add the one with the greatest WCA value. Before adding the next $K-1$ edges, we move the reparameterized edge potential into the HOP, recompute the argmins over $\tilde{\theta}_\alpha$, then update the WCAs. Notice that the WCA measure relates to the weak tree agreement (WTA) measure from Tarlow et al. (2011). We do not use the WCA to select the starting set of edges because we have found that in many times after $\mathbf{LP}_\emptyset$ converges, the WCA of all the edges is equal to zero, i.e., there is no single edge whose addition to S will tighten the bound. This rarely happens when S is non-empty.

8 Experimental Results

We conducted three sets of experiments over graphical models with different kinds of cardinality HOPs. The first extends the first example in Sec. 2 and shows experimentally that $\mathbf{LP}_\emptyset$ does not find the MAP solution in many simple cases. The second compares our edge selection criterion from Sec. 7 with other possible criteria and shows it is superior to them. The last set of experiments was done over images from the Berkeley segmentation dataset (Martin et al., 2001) and shows we can often find the optimal average-cut (NP-hard problem in the general case) by solving $\mathbf{LP}_S$ with a low tree-width set of edges. The first set of experiments was done on a relatively small problem which allowed us to use a commercial LP solver (Mosek). Since the goal of this experiment was to understand $\mathbf{LP}_\emptyset$ we preferred using it and avoid possible difficulties when solving the dual problem. The other two experiments were solved using message passing with convex belief propagation, as described in (Weiss et al., 2007, 2011), applied to $\mathbf{LP}_S$. We built our junction-tree code on top of the UGM package (Schmidt, 2012).



(a)



(b)

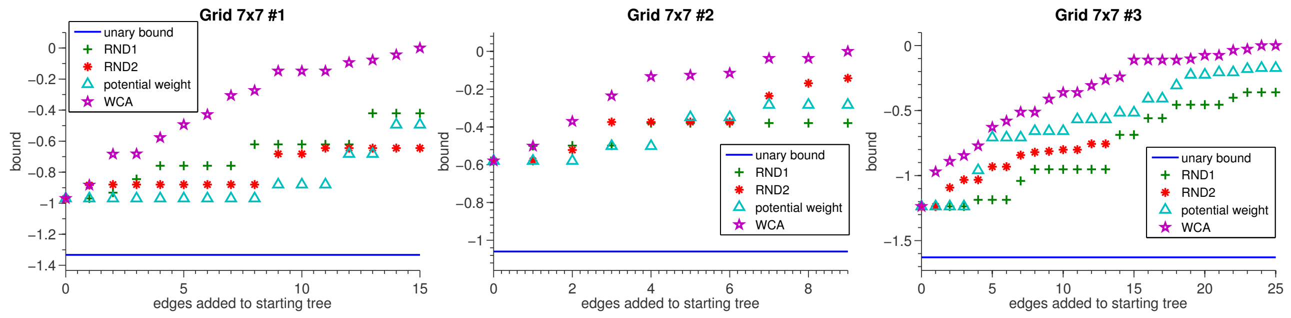
Figure 8.1: $\mathbf{LP}_{\emptyset}$ Hamming ball exclusion. The 2nd best mode of a tree model with attractive pairwise potentials (λ) is rarely found when using only unary consistency constraints ($\mathbf{LP}_{\emptyset}$). We conducted the experiments for several radiuses of hamming balls (k). Using our suggested method it can always be found exactly and efficiently.

Hamming Ball Exclusion When finding M-best modes (Batra et al., 2012), it is desirable to have dissimilarity measures that do not factorize into a sum of pixel dissimilarities, e.g., to represent the notion of *dissimilar enough*, where as long as an assignment is at least Hamming distance k away, we are equally happy. In this case, the constraint of Fromer and Globerson (2009) is no longer applicable, so the strategy suggested by Batra et al. (2012), is to add a HOP enforcing this constraint, then solving $\mathbf{LP}_{\emptyset}$. We know from our theoretical analysis that $\mathbf{LP}_{\emptyset}$ cannot solve all instances of this problem. Here, we empirically study how bad the relaxation becomes (measured by the integrality gap) as the strength of pairwise potentials is varied. We generated random trees over 10 variables with attractive pairwise potentials $\theta_{ij}(x_i, x_j) = \begin{pmatrix} 0 & \lambda \\ \lambda & 0 \end{pmatrix}$, and unary potentials with random preference to be off ($\theta_i(1) \sim \mathcal{U}[0,1]$); thus the MAP assignment is all zeros. We add an exclusion factor which allows only assignments which are at least k Hamming distance away: $\theta_{\alpha}(\mathbf{x}) = \infty$ if $|x| < k$ and 0 otherwise. Fig. 8.1 shows the percent of integral solutions and the average integrality gap (out of 100 experiments) for different k's and $\lambda's$. Notice that while $\mathbf{LP}_{\emptyset}$ fails in finding the MAP for this problem, the tree-width of $\mathbf{LP}_S$ here is one and thus we can solve it efficiently and exactly.

Sequentially Adding Edges We compare the improvements in the dual bound that result from using different criteria for choosing edges to add to S and then solving $\mathbf{LP}_S$. The first criterion is our suggested WCA method, and the second is the potential weight heuristic, both of which are described in Sec. 7. The third criterion is simply adding random edges (RND1 and RND2). We conducted experiments over several 4-connected 7x7 grid, with random pairwise potentials. The HOP is average-cut, $\theta_{\alpha}(\mathbf{x}) = -\lambda \cdot |\mathbf{x}| \cdot (n - |\mathbf{x}|)$, where λ is chosen such that the optimal energy will be zero, and $\theta_1(0) = \infty$ to break the symmetry. We did not limit the tree-width in this experiment. Figure 8.2 shows the improvement in the bound after each edge addition to the starting tree. Clearly the WCA criterion is better than the other two baselines.

Image Segmentation using super-pixels Finally, we construct average-cut problems for 40 images from the Berkeley segmentation dataset (Martin et al., 2001) and attempt to solve them using $\mathbf{LP}_S$ and the WCA measure for choosing S. We used the procedure described in (Mezuman and Weiss, 2012) to find a setting of λ such that solving $\mathbf{LP}_{\mathcal{E}}$ would verify the optimality of the average-cut. We used SLIC superpixels (Achanta et al., 2012) using the implementation of (Mueller, 2012). We chose SLIC parameters to get approximately 100 equally sized superpixels. The pairwise potentials (affinities between pixels) were computed using intervening contours (Leung and Malik, 1998) (implementation provided by (Cour et al., 2010)).

When choosing S, we limited the tree-width to be at most six (the average tree-width of the full graph is 13). We add edges in batches of eight after the previous $\mathbf{LP}_S$ was solved (i.e., the BP converged). Via this procedure we provably found the optimal solution in 34 out of the 40 images. For 2 images the optimum was found when S was of tree-width of 2, for 13 when S was of tree-width 3, and for 10, 5 and 4 images when S was of tree-width of 4, 5 and 6, respectively. Our solution improved the standard spectral solution in 38 out of 40 problems, with an average improvement in the objective of 70%. Fig. 8.4 shows the maximal dual bound achieved during message passing across sets of edges with different tree-widths. To keep the plot clean we show it only for the 25 images for which the $\mathbf{LP}_S$ was tight with tree-width at most four.



(a) (b) (c)

Figure 8.2: Sequentially adding edges. Comparison of the improvement in the dual bound after sequential edge addition using different criteria.

9 Discussion

Recent years have shown a resurgence of interest in higher-order potentials for graphical models with a growing number of specific potentials for which message-passing algorithms can be applied efficiently. In this paper we have shown that many of these methods are based on a particular linear programming relaxation and highlighted the weakness of that relaxation. We suggested a family of tighter relaxations which result in a practical new strategy that can yield significantly improved accuracy. The computational cost incurred for this increased accuracy is exponential in the tree-width of the consistency sets S, but empirically we see that substantial gains in accuracy can be achieved with relatively small tree-width.

One challenge for the cardinality HOP is scaling up to bigger problems. Regardless of the tree-width, there is a computational cost to the message computations that is quadratic in the number of variables in the model. When we apply our approach to large problems (e.g., images where individual pixels are variables), this cost becomes a bottleneck. A strategy that we would like to investigate is to use Fast Fourier Transforms near the zero-temperature limit to compute approximate max-marginals, or to investigate other algorithms for fast approximate max-convolution. Throughout, we assumed that the variables are binary and that the energy decomposes to unary and pairwise potentials with a single HOP. Extensions to the 3rd or 4th order cliques that are commonly used in computer vision would be straightforward by enforcing consistency between the HOP and all these cliques, and the assumption of binary variables can be removed so long as the HOP computations can be done tractably. For the case of multiple HOPs in the model, while there are open questions about some of the specifics (e.g., should different HOPs be constrained to choose the same sets of edges or not?), the same basic approach that we presented here is also applicable, and we believe it to be a good choice. Finally, we have shown how to efficiently compute message updates for two classes of HOPs. An open question and a new computational challenge is to discover other cases where messages can similarly be computed efficiently.

Acknowledgements

This research is partially supported by the ISF Centers of Excellence grant 1789/11, CIfAR and the Charitable Gatsby Foundation.

A Proof that $\mathbf{LP}_\emptyset = \mathbf{LP}_{\mathbf{red}}$

PRIMAL $\mathbf{LP}_\emptyset \leq$ PRIMAL $\mathbf{LP}_{\mathbf{red}}$ Copy the solution from $\mathbf{LP}_{\mathbf{red}}$ into $\mathbf{LP}_\emptyset$ for the variables that correspond, and set $q_\alpha(\mathbf{x}) = \sum_z \frac{\prod_{i'} q_{zi'}(z,x_{i'})}{q_z(z)^{n-1}}$, which is a distribution over x and z that has $q_{zi}(z,x_i)$ as its pairwise marginals, because this corresponds to the Bethe approximation on a tree-structured graph (in this case, a star around z), which is exact. Unary consistency in $\mathbf{LP}_\emptyset$ is satisfied because

$$\sum_{\mathbf{x}:\mathbf{x}(i)=x_i} q_\alpha(\mathbf{x}) = \sum_{\mathbf{x}:\mathbf{x}(i)=x_i} \sum_z \frac{\prod_{i'} q_{zi'}(z,x_{i'})}{q_z(z)^{n-1}} \tag{A.1}$$

$$= \sum_z q_{zi}(z,x_i) = q_i(x_i). \tag{A.2}$$

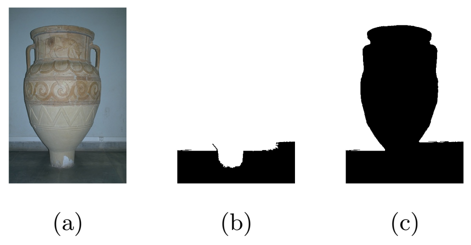
The objective of $\mathbf{LP}_\emptyset$ is less than or equal to the objective at $\mathbf{LP}_{\mathbf{red}}$, because $\theta_\alpha(\mathbf{x}) = \min_z \sum_i \theta_{zi}(z,x_i)$:

$$\sum_{\mathbf{x}} q_\alpha(\mathbf{x})\theta_\alpha(\mathbf{x}) \tag{A.3}$$

$$= \sum_{\mathbf{x}} (\sum_z \frac{\prod_{i'} q_{zi'}(z,x_{i'})}{q_z(z)^{n-1}})(\min_z \sum_i \theta_{zi}(z,x_i)) \tag{A.4}$$

$$\leq \sum_{\mathbf{x}} \sum_z \frac{\prod_{i'} q_{zi'}(z,x_{i'})}{q_z(z)^{n-1}} \sum_i \theta_{zi}(z,x_i) \tag{A.5}$$

$$= \sum_i \sum_{z,x_i} q_{zi}(z,x_i)\theta_{zi}(z,x_i). \tag{A.6}$$



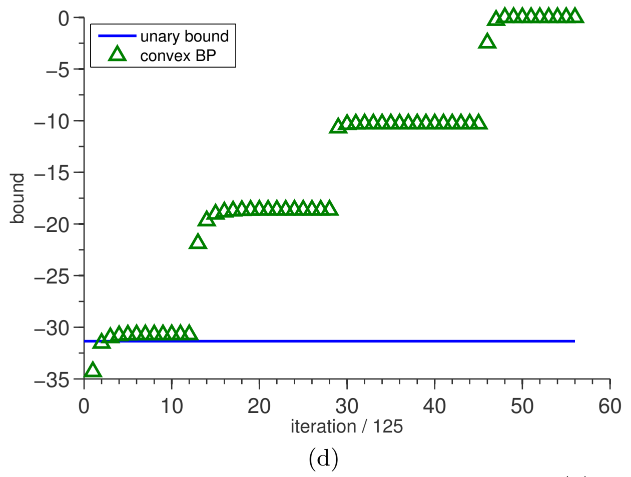


Figure 8.3: Image Segmentation example run. (a) Input image. (b) Suboptimal solution found by the commonly used spectral method. (c) The global optimum, which is found and verified by our algorithm. (d) The dual bound of convex BP versus iteration of message passing. Jumps occur when edges are added to S.

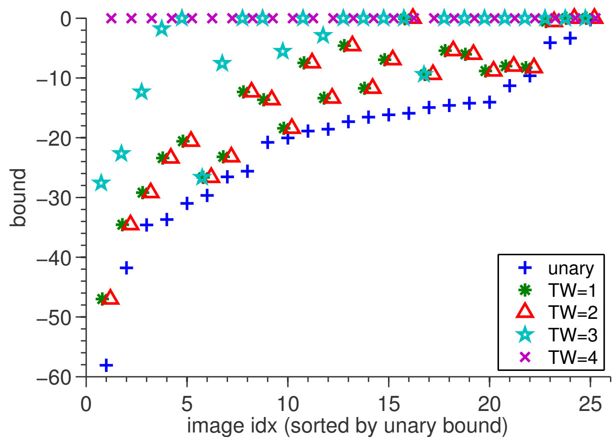


Figure 8.4: Lower bounds achieved by our algorithm versus the standard $\mathbf{LP}_\emptyset$ bound ("unary") on average-cut image segmentation problems at various tree-widths of S, ranging from 1 to 4 ("TW-1" to "TW-4"). In all cases the optimal integral energy is 0, so when the lower bound reaches 0 we have provably reached the optimum.

DUAL $\mathbf{LP}_\emptyset \geq$ DUAL $\mathbf{LP}_{\mathbf{red}}$ Here we take a dual solution from $\mathbf{LP}_{\mathbf{red}}$ and construct a dual objective for $\mathbf{LP}_\emptyset$ that is greater than or equal to the DUAL$_{red}$ objective. For a given setting of dual variables δ and γ, the duals for $\mathbf{LP}_\emptyset$ (DUAL$_\emptyset$) and $\mathbf{LP}_{\mathbf{red}}$ (DUAL$_{red}$), respectively, are as follows:

DUAL$_{red}$

$$\sum_i \min_{x_i}[\theta_i(x_i) + \sum_{j\in N(i)} \delta_{ji}(x_i) + \gamma_{zi}(x_i)] \tag{A.7}$$
$$+ \sum_{ij} \min_{x_i,x_j}[\theta_{ij}(x_i,x_j) - \delta_{ji}(x_i) - \delta_{ij}(x_j)]$$
$$+ \sum_i \min_{z,x_i}[\theta_{zi}(z,x_i) - \gamma_{zi}(x_i) - \gamma_{iz}(z)] + \min_z[\sum_i \gamma_{iz}(z)]$$

DUAL$_\emptyset$

$$\sum_i \min_{x_i}[\theta_i(x_i) + \sum_{j\in N(i)} \delta_{ji}(x_i) + \gamma_i(x_i)] \tag{A.8}$$
$$+ \sum_{ij} \min_{x_i,x_j}[\theta_{ij}(x_i,x_j) - \delta_{ji}(x_i) - \delta_{ij}(x_j)]$$
$$+ \min_{\mathbf{x}}[\theta_\alpha(\mathbf{x}) - \sum_i \gamma_i(x_i)]$$

Now copy the δ messages from DUAL$_{red}$ to DUAL$_\emptyset$. In DUAL$_\emptyset$, set $\gamma_i(x_i) = \gamma_{zi}(x_i)$. Then the difference in the dual objectives between Eq. A.8 and Eq. A.7 is only in the terms involving z or α. Focusing on the z terms from Eq. A.7:

$$\sum_i \min_{z,x_i}[\theta_{zi}(z,x_i) - \gamma_{zi}(x_i) - \gamma_{iz}(z)] + \min_z[\sum_i \gamma_{iz}(z)] \tag{A.9}$$
$$\leq \min_{z,\mathbf{x}} \sum_i [\theta_{zi}(z,x_i) - \gamma_{zi}(x_i) - \gamma_{iz}(z)] + \min_z[\sum_i \gamma_{iz}(z)] \tag{A.10}$$
$$\leq \min_{z,\mathbf{x}} \sum_i [\theta_{zi}(z,x_i) - \gamma_{zi}(x_i)] \tag{A.11}$$
$$= \min_{\mathbf{x}}[\theta_\alpha(\mathbf{x}) - \sum_i \gamma_i(x_i)], \tag{A.12}$$

which shows that the dual objective for $\mathbf{LP}_\emptyset$ is greater than or equal to the dual objective for $\mathbf{LP}_{\mathbf{red}}$ with this choice.

So we have DUAL$_\emptyset$ $\geq$ DUAL$_{red}$ and PRIMAL $\mathbf{LP}_{\mathbf{red}} \geq$ PRIMAL $\mathbf{LP}_\emptyset$. Since strong duality for LPs gives DUAL$_\emptyset$ = PRIMAL $\mathbf{LP}_\emptyset$ and DUAL$_{red}$ = PRIMAL $\mathbf{LP}_{\mathbf{red}}$, this implies that the two LPs have the same solution value and are thus equally tight.

References

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *TPAMI*, 2012.

D. Batra, S. Nowozin, and P. Kohli. Tighter relaxations for MAP-MRF inference: A local primal-dual gap based separation algorithm. In *AISTATS*, 2011.

D. Batra, P. Yadollahpour, A. Guzman-Rivera, and G. Shakhnarovich. Diverse M-best solutions in Markov Random Fields. *ECCV*, 2012.

C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in bayesian networks. In *UAI*, 1996.

T. Cour, S. Yu, and J. Shi. Matlab normalized cuts segmentation code. `www.seas.upenn.edu/~timothee/software/ncut/ncut.html`, 2010.

M. Fromer and A. Globerson. An LP view of the M-best MAP problem. *NIPS*, 2009.

A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. *NIPS*, 2007.

S. Gould. Max-margin learning for lower linear envelope potentials in binary Markov Random Fields. In *ICML*, 2011.

D. Heckerman. A tractable inference algorithm for diagnosing multiple diseases. In *UAI*, 1989.

P. Kohli, M. Kumar, and P. Torr. P3 & beyond: Solving energies with higher order cliques. In *CVPR*, 2007.

P. Kohli, L. Ladicky, and P. H. Torr. Robust higher order potentials for enforcing label consistency. *IJCV*, 2009.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, 2009.

V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *TPAMI*, 2006.

N. Komodakis and N. Paragios. Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles. 2008.

N. Komodakis and N. Paragios. Beyond pairwise energies: Efficient optimization for higher-order MRFs. In *CVPR*, 2009.

N. Komodakis, N. Paragios, and G. Tziritas. Mrf optimization via dual decomposition: Message-passing revisited. 2007.

N. Komodakis, N. Paragios, and G. Tziritas. Mrf energy minimization and beyond via dual decomposition. *TPAMI*, 2010.

T. Koo, A. Rush, M. Collins, T. Jaakkola, and D. Sontag. Dual decomposition for parsing with non-projective head automata. In *ACL-EMNLP*, 2010.

T. Leung and J. Malik. Contour continuity in region based image segmentation. *ECCV*, 1998.

D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, 2001.

A. Martins, N. Smith, E. Xing, P. Aguiar, and M. Figueiredo. Turbo parsers: Dependency parsing by approximate variational inference. In *ACL-EMNLP*, 2010.

E. Mezuman and Y. Weiss. Globally optimizing graph partitioning problems using message passing. In *AISTATS*, 2012.

A. Mueller. Superpixels for python - pretty SLIC, 2012.

S. Nowozin and C. Lampert. Global connectivity potentials for random field models. In *CVPR*, 2009.

J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann, 1988.

C. Rother, P. Kohli, W. Feng, and J. Jia. Minimizing sparse higher order energy functions of discrete variables. In *CVPR*, 2009.

E. Santos Jr. On the generation of alternative explanations with implications for belief revision. In *UAI*, 1991.

M. Schmidt. Ugm: matlab code for undirected graphical models. `www.di.ens.fr/~mschmidt/Software/UGM.html`, 2012.

D. Smith and J. Eisner. Dependency parsing by belief propagation. In *ACL-EMNLP*, 2008.

D. Sontag. *Approximate Inference in Graphical Models using LP Relaxations.* PhD thesis, MIT, EECS, 2010.

D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. 2008.

D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. 2010.

D. Sontag, D. K. Choe, and Y. Li. Efficiently searching for frustrated cycles in MAP inference. In *UAI*, 2012.

K. Swersky, D. Tarlow, I. Sutskever, R. Salakhutdinov, R. Zemel, and Adams. Cardinality Restricted Boltzmann Machines. In *NIPS*, 2012.

D. Tarlow, I. Givoni, and R. Zemel. HOP-MAP: Efficient message passing with high order potentials. In *AISTATS*, 2010.

D. Tarlow, D. Batra, P. Kohli, and V. Kolmogorov. Dynamic tree block coordinate ascent. In *ICML*, 2011.

D. Tarlow, K. Swersky, R. Zemel, R. Adams, and B. Frey. Fast exact inference for recursive cardinality models. In *UAI*, 2012.

S. Vicente, V. Kolmogorov, and C. Rother. Joint optimization of segmentation and appearance models. In *ICCV*. 2009.

M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 2008.

Y. Weiss, C. Yanover, and T. Meltzer. MAP estimation, linear programming and belief propagation with convex free energies. In *UAI*. Citeseer, 2007.

Y. Weiss, C. Yanover, and T. Meltzer. *Linear Programming and variants of Belief Propagation in (Blake and Rother, ed.) Markov Random Fields for Vision and Image Processing.* Mit Pr, 2011.

T. Werner. A linear programming approach to max-sum problem: A review. *TPAMI*, 2007.

T. Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (map-mrf). In *CVPR*, 2008.

J. Yedidia, W. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. on Info. Theory*, 2005.

Cyclic Causal Discovery from Continuous Equilibrium Data

Joris M. Mooij[*]
Informatics Institute
University of Amsterdam
The Netherlands

Tom Heskes
Institute for Computing and Information Sciences
Radboud University Nijmegen
The Netherlands

Abstract

We propose a method for learning cyclic causal models from a combination of observational and interventional equilibrium data. Novel aspects of the proposed method are its ability to work with continuous data (without assuming linearity) and to deal with feedback loops. Within the context of biochemical reactions, we also propose a novel way of modeling interventions that modify the *activity* of compounds instead of their abundance. For computational reasons, we approximate the nonlinear causal mechanisms by (coupled) local linearizations, one for each experimental condition. We apply the method to reconstruct a cellular signaling network from the flow cytometry data measured by Sachs et al. (2005). We show that our method finds evidence in the data for feedback loops and that it gives a more accurate quantitative description of the data at comparable model complexity.

1 Introduction

A central question that arises in many empirical sciences is how to discover cause-effect relationships between variables from measured data. Knowledge of causal relationships is essential in order to predict how a system will react to interventions that perturb the system from its natural state (Pearl, 2000), which is very useful for many practical applications. An example from biology is the problem of predicting *in silico* how a signaling pathway in a cell will react *in vitro* when it is treated with a certain chemical compound. The ability to reliably make such causal predictions can be a powerful tool for practical applications like drug design.

A concrete example is the multivariate proteomics data set measured and analyzed by Sachs et al. (2005). Using flow cytometry, abundances of 11 biochemical compounds (phosphorylated proteins and phospholipid components) were measured in single human immune system cells under various experimental perturbations. Sachs et al. (2005) reconstructed the underlying "signaling network" by learning Bayesian networks from the data. Their reconstruction turned out to be very close to the "well-established consensus network" that had been obtained by manually combining results from many different experiments, an effort that had taken about two decades.

The consensus network contains 18 expected causal relationships. Sachs et al. (2005) found two new unexpected causal relations (and experimentally verified one of them) and obtained one reversed relationship. However, they did not recover three of the 18 expected causal relationships. Sachs et al. (2005) hypothesized that these three missing causal relationships are all involved in feedback loops. As Bayesian networks are acyclic by definition, this could explain why they were not found by their method. Additional support for this hypothesis can be found by simple inspection of the data, which already shows strong evidence for the presence of feedback loops. In later work (Itani et al., 2010), the authors proposed a heuristic method for causal discovery that takes into account the possibility of feedback. An alternative approach to dealing with cycles was proposed by Schmidt and Murphy (2009).

A common feature of the causal discovery methods that have been applied so far on this protein data set is that they all work with *discretized* data: although the raw measurements are continuous-valued, the data is preprocessed by discretization into three coarse categories (low, medium and high abundance). We argue that discretization of the data as a preprocessing step should be avoided if possible, as this throws away much

[*]Also affiliated to Institute for Computing and Information Sciences, Radboud University Nijmegen, The Netherlands

of the information in the data that could be useful for causal discovery. Recently, several methods have been proposed for causal discovery from continuous-valued observational data by exploiting independence of the estimated noise with the input (Shimizu et al., 2006; Hoyer et al., 2009; Zhang and Hyvärinen, 2009; Peters et al., 2011). Similar ideas have also been studied in the cyclic case (Lacerda et al., 2008; Mooij et al., 2011). More recently, cyclic methods that can deal with hidden common causes and with a combination of observational and experimental data have been proposed (Eberhardt et al., 2010; Hyttinen et al., 2012).

However, none of these methods are directly applicable to the (Sachs et al., 2005) data set, among others because they model interventions in a different way. Most interventions performed by Sachs et al. (2005) change the *activity* of a compound, not its *abundance*, and therefore the standard formalism for interventions (Pearl, 2000) is not applicable. Sachs et al. (2005) and Itani et al. (2010) propose two different ways of modeling these interventions that both exploit the fact that the data has been discretized. Eaton and Murphy (2007) consider different possible intervention types and learn the interventions from the data, instead of using the biological background knowledge in (Sachs et al., 2005). Eaton and Murphy (2007) conclude that the data can be best explained by assuming that most interventions are not as specific as originally assumed by Sachs et al. (2005), but act on multiple compounds simultaneously (also known as "fat-hand" interventions). In this work, we offer an alternative explanation, where we assume that interventions are specific (i.e., act only on a single compound), but where most interventions change the *activity* of that compound (i.e., the way in which it influences the equilibrium distributions of its direct effects). In addition, feedback loops may increase the impact of an intervention.

The goal of this work is to develop a practical method for analyzing data sets such as the protein data collected by Sachs et al. (2005). The method we propose here does not start by throwing away information (by discretizing the data as a preprocessing step), but works directly with the original continuous-valued measurements. As we expect feedback loops to play a prominent role in biological networks, we also drop the assumption of acyclicity. Another feature of our method that distinguishes it from many existing approaches is that we will not assume linearity of the causal mechanisms but allow for nonlinearities. Finally, we propose a natural and in our opinion more realistic way of modeling activity interventions.

2 Modeling assumptions

In this section we describe our modeling assumptions in detail. First of all, the data form a "snapshot" of a dynamical process: for each individual cell we have one multivariate measurement done at a single point in time. Therefore, we will assume that the cells have reached *equilibrium* when the measurements are performed (an assumption called "homeostasis" in biology). This is an approximation, but a necessary one in the light of the absence of time-series data.

2.1 Structural Causal Models

We will assume that the equilibrium data can be described by a *Structural Causal Model (SCM)* (Pearl, 2000), also known as *Structural Equation Model (SEM)* (Bollen, 1989). In particular, for D observed variables $x_1, \dots, x_D$ (corresponding in our case to the abundances of the biochemical compounds), the model consists of D *structural equations*

$$x_i = f_i(\boldsymbol{x}_{\mathrm{pa}(i)}, \epsilon_i) \qquad i = 1, \dots, D \tag{1}$$

where $\mathrm{pa}(i) \subseteq \{1, \dots, D\} \setminus \{i\}$ is the set of *parents* (direct causes) of x_i, f_i is the *causal mechanism* determining the value of the effect x_i in terms of its direct causes $\boldsymbol{x}_{\mathrm{pa}(i)}$ and a *disturbance variable* ϵ_i representing all unobserved other causes of x_i. In addition, an SCM specifies a joint probability density $p(\boldsymbol{\epsilon})$ on the disturbance variables $\epsilon_1, \dots, \epsilon_D$. Following Sachs et al. (2005), we will make the assumption of *causal sufficiency*, which means that we exclude the possibility of *confounders* (i.e., hidden common causes of two or more observed variables). In other words, we assume that the disturbance variables are jointly independent: $p(\boldsymbol{\epsilon}) = \prod_{i=1}^{n} p(\epsilon_i)$. Without loss of generality, we will additionally assume that $\mathbb{E}(\boldsymbol{\epsilon}) = \mathbf{0}$ and $\mathbb{V}\mathrm{ar}(\boldsymbol{\epsilon}) = \boldsymbol{I}$.

The structure of a causally sufficient SCM $\mathcal{M}$ can be visualized with a directed graph $\mathcal{G}_{\mathcal{M}}$ with vertices $\{x_1, \dots, x_D\}$ and edges $x_j \to x_i$ if and only if f_i depends on x_j, i.e., if $j \in \mathrm{pa}(i)$. As we do not exclude the possibility of feedback loops, the graph $\mathcal{G}_{\mathcal{M}}$ is not necessarily acyclic, but may contain directed cycles. We will assume that for each joint value $\boldsymbol{\epsilon}$ of the disturbance variables, there exists a unique solution $\boldsymbol{x}(\boldsymbol{\epsilon})$ of the D structural equations (1). Note that this assumption is automatically satisfied in the acyclic case, but that it induces additional constraints in the cyclic case. This assumption implies that the distribution $p(\boldsymbol{\epsilon})$ induces a distribution $p(\boldsymbol{x})$ on the observed variables. This induced distribution is called the *observational distribution* of the SCM. In addition, we will assume that the mapping $\boldsymbol{\epsilon} \mapsto \boldsymbol{x}$ is invertable. In that case, the observational density is given explicitly by:

$$p(\boldsymbol{x}) = p\big(\boldsymbol{\epsilon}(\boldsymbol{x})\big)\left|\det\frac{d\boldsymbol{\epsilon}}{d\boldsymbol{x}}\right| = \left(\prod_{i=1}^{D} p(\epsilon_i)\right)\left|\det\frac{d\boldsymbol{\epsilon}}{d\boldsymbol{x}}\right|. \quad (2)$$

2.2 Interventions

The SCM literature typically considers "perfect interventions", which are modeled as follows (Pearl, 2000). Under an intervention "$\mathrm{do}(x_i = \xi_i)$" that forces the variable x_i to attain the value ξ_i, the SCM is adapted by replacing the structural equation for x_i with the equation $x_i = \xi_i$, while leaving the other aspects of the SCM invariant. In particular, the distribution on the disturbance variables $p(\boldsymbol{\epsilon})$ stays the same; however, because one of the structural equations changed, the induced distribution on the observed variables $\boldsymbol{x}$ changes into the *interventional* distribution, with density $p\big(\boldsymbol{x} \mid \mathrm{do}(x_i = \xi_i)\big)$. In the cyclic case, we also need to assume that under the relevant interventions, there exists a unique solution $\boldsymbol{x}(\boldsymbol{\epsilon})$ of the (modified) structural equations for each value of $\boldsymbol{\epsilon}$; otherwise, the induced (interventional) distribution will be ill-defined.

These "perfect interventions" correspond in the case of the signaling network data with interventions that change the *abundance* of a compound. However, many of the interventions actually performed by (Sachs et al., 2005) do not directly change the abundance, but rather its *activity*, i.e., the extent to which it influences abundances of other compounds. In their original paper, (Sachs et al., 2005) model these "activity interventions" in the following way: if the activity of compound i is *inhibited*, the actual measurements of x_i are replaced with the value "low", whereas if compound i is *activated*, the actual measurements of x_i are replaced with the value "high". Not only does this approach throw away data, it also depends on the discretization of the data. In later work, (Itani et al., 2010) model these interventions in a different way: they split the variable x_i into two parts, x_i and x_i^{int}, where x_i^{int} is assigned the value corresponding to the intervention (either "low" in case of an inhibitor or "high" in case of an activator), and x_i represents the abundance of compound i measured in the interventional experiment. In the modified graph corresponding to the intervention, the outgoing arrows from x_i now become outgoing arrows of x_i^{int} instead, and all incoming arrows go into x_i. This approach no longer throws away data, but it still requires a coarse discretization of the data.

Instead, we propose to model these activity interventions as follows: if an intervention changes the *activity* of compound i, we adapt the SCM by allowing the *children*[1] j of compound i to change their causal mechanism $f_j(\boldsymbol{x}_{\mathrm{pa}(j)}, \epsilon_j)$ into a different function $\tilde{f}_j(\boldsymbol{x}_{\mathrm{pa}(j)}, \epsilon_j)$, whereas the other aspects of the SCM (including its structure) remain invariant. In our context, this new causal mechanism $\tilde{f}_j$ is unknown and we learn it from the data. In particular, we do not use the background knowledge provided by Sachs et al. (2005) that specifies whether an activity intervention is an inhibitor or an activator.

[1]The children of i are all j such that $i \in \mathrm{pa}(j)$.

2.3 Approximating causal mechanisms

So far, we have not assumed linearity, and in theory we could proceed by modeling the causal mechanisms as nonparametric nonlinear functions, e.g., as Gaussian Processes (Rasmussen and Williams, 2006). For computational reasons, however, we linearize the causal mechanisms in the SCM locally around their average input $(\langle \boldsymbol{X}_{\mathrm{pa}(i)} \rangle, 0)$:[2]

$$f_j(\boldsymbol{x}_{\mathrm{pa}(j)}, \epsilon_j) \approx \sum_{i=1}^{D} B_{ij} x_i + \mu_j + \alpha_j \epsilon_j$$

where we introduced the matrix $\boldsymbol{B} \in \mathbb{R}^{D\times D}$ and vectors $\boldsymbol{\mu}, \boldsymbol{\alpha} \in \mathbb{R}^{D\times 1}$, defined by:

$$B_{ij} := \left.\frac{\partial f_j}{\partial x_i}\right|_{(\langle \boldsymbol{X}_{\mathrm{pa}(j)}\rangle, 0)}, \qquad \alpha_j := \left.\frac{\partial f_j}{\partial \epsilon_j}\right|_{(\langle \boldsymbol{X}_{\mathrm{pa}(j)}\rangle, 0)},$$

$$\mu_j := f_j\big(\langle \boldsymbol{X}_{\mathrm{pa}(j)}\rangle, 0\big) - \sum_{i\in \mathrm{pa}(j)} \left.\frac{\partial f_j}{\partial x_i}\right|_{(\langle \boldsymbol{X}_{\mathrm{pa}(j)}\rangle, 0)} \langle X_i \rangle.$$

Note that the structure of the matrix $\boldsymbol{B}$ reflects the graph structure $\mathcal{G}_{\mathcal{M}}$ of the model: it has zeroes on the diagonal, and B_{ij} is the (linearized) direct effect of x_i on x_j, which can only be nonzero if $i \in \mathrm{pa}(j)$.

This means that for a single experimental condition, we assume the following linearized structural equations:

$$x_j = \boldsymbol{x}^T \boldsymbol{B}_{\cdot j} + \mu_j + \alpha_j \epsilon_j.$$

For an i.i.d. sample of N data points arranged in the matrix $\boldsymbol{X} \in \mathbb{R}^{N\times D}$ and latent disturbance variables $\boldsymbol{E} \in \mathbb{R}^{N\times D}$, these can be written in matrix notation:

$$\boldsymbol{X}(\boldsymbol{I} - \boldsymbol{B}) = \boldsymbol{1}_N \boldsymbol{\mu}^T + \boldsymbol{E}\boldsymbol{\alpha}^T. \quad (3)$$

Under some upstream intervention, the average input $(\langle \boldsymbol{X}_{\mathrm{pa}(i)}\rangle, 0)$ of the causal mechanism f_i for compound i may change. If the change is large with respect to the curvature of f_i, we may need to relinearize f_i around the new average input under the intervention (even though the nonlinear function f_i itself may have remained unchanged, see also Figure 2(a)). This will be discussed in detail in section 2.5.

[2]We denote the empirical mean of a variable x by $\langle X \rangle$.

$$\begin{tikzcd}
 & x_1 \arrow[dl] \arrow[dr] & \\
x_2 \arrow[dr] & & x_3 \arrow[dl] \\
\mathcal{G} & x_4 &
\end{tikzcd}$$

c	Intervention type
1	none (observational)
2	activity of x_1
3	activity of x_2
4	abundance of x_3
5	abundance of x_1

$$m_{ic}(\mathcal{G}) = \begin{pmatrix} 1 & 1 & 1 & 1 & 2 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 3 & 1 \\ 1 & 1 & 2 & 1 & 1 \end{pmatrix} \qquad M_i(\mathcal{G}) = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 2 \end{pmatrix}$$

Figure 1: Example of a graph $\mathcal{G}$, experimental metadata, and the corresponding mechanism labels $m_{ic}(\mathcal{G})$. As an example, $m_{32}(\mathcal{G}) = 2$ because the second experimental condition, an activity intervention on x_1, changes the causal mechanism of x_3. In the third condition, the causal mechanism of x_3 is identical to that in the first condition, so $m_{33}(\mathcal{G}) = m_{31}(\mathcal{G}) = 1$.

2.4 Likelihood

Assuming that all disturbance variables ϵ_i have the same probability density $p(\epsilon_i) = p_0$, the likelihood of i.i.d. data $\boldsymbol{X}$ for a single experimental condition follows directly from expressions (2) and (3):

$$p(\boldsymbol{X} \mid \boldsymbol{B}, \boldsymbol{\mu}, \boldsymbol{\alpha}) = \prod_{n=1}^{N} \Bigg[|\det(\boldsymbol{I} - \boldsymbol{B})| \cdot \prod_{i=1}^{D} \frac{1}{\alpha_i} p_0 \left(\frac{(\boldsymbol{X}(\boldsymbol{I} - \boldsymbol{B}))_{ni} - \mu_i}{\alpha_i} \right) \Bigg]. \tag{4}$$

We will consider two choices for the noise density, Gaussian noise $p_0(e) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}e^2)$ and super-Gaussian noise that is often used in the Independent Component Analysis (ICA) literature, $p_0(e) = 1/\big(\pi \cosh(e)\big)$. Note that in the acyclic case, $\det(\boldsymbol{I} - \boldsymbol{B}) = 1$, and therefore the likelihood factorizes over variables. This simplification does not occur in the cyclic case, as the likelihoods of different variables may become coupled through the determinant.

Combining data from different experimental conditions $c = 1, \dots, K$ is straightforward:

$$p\big((\boldsymbol{X})_{c=1}^{K} \,|\, (\boldsymbol{B}^{(c)}, \boldsymbol{\mu}^{(c)}, \boldsymbol{\alpha}^{(c)})_{c=1}^{K}\big) = \prod_{c=1}^{K} p(\boldsymbol{X}^{(c)} \,|\, \boldsymbol{B}^{(c)}, \boldsymbol{\mu}^{(c)}, \boldsymbol{\alpha}^{(c)}),$$

where the superscript "(c)" labels data and parameters corresponding to the c'th experimental condition.

2.5 Parameter priors

We denote all parameters of the linearized SCM corresponding to experimental condition c by $\boldsymbol{\Theta}^{(c)} := (\boldsymbol{B}^{(c)}, \boldsymbol{\mu}^{(c)}, \log \boldsymbol{\alpha}^{(c)})$, and the subset of parameters corresponding only to the causal mechanism $f_i^{(c)}$ of the i'th compound as $\boldsymbol{\Theta}_i^{(c)} := (B_{\cdot i}^{(c)}, \mu_i^{(c)}, \log \alpha_i^{(c)})$. Using the Bayesian approach to multi-task learning, we couple these K learning problems by imposing a prior $p\big(\boldsymbol{\Theta}^{(1)}, \dots, \boldsymbol{\Theta}^{(K)} \,|\, \mathcal{G}\big)$ on the parameters that embodies our assumption that these parameters should be related in specific ways, conditional on the hypothetical causal structure $\mathcal{G}$ of the SCM. The hypothetical structure $\mathcal{G}$ always constrains the structure of $\boldsymbol{B}$ in the sense that if $i \notin \mathrm{pa}_{\mathcal{G}}(j)$, $B_{ij}^{(c)} = 0$ for all $c = 1, \dots, K$. We consider various choices to couple the nonzero parameters of the $\boldsymbol{B}^{(c)}$, and the location and scale parameters $\boldsymbol{\mu}^{(c)}$ and $\boldsymbol{\alpha}^{(c)}$, across experimental conditions.

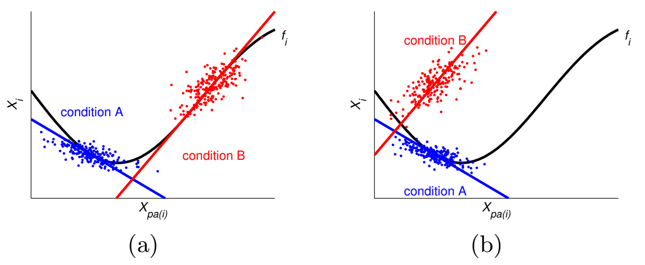


Figure 2: (a) Even though some causal mechanism $f_i(\boldsymbol{x}_{\mathrm{pa}(i)}, \epsilon_i)$ stays invariant, its linearization around the new equilibrium may have changed. (b) In this case, even a nonlinear causal mechanism cannot fit the data well. Another structure, assigning different causal mechanisms to conditions A and B, may better fit the data.

For each compound i, the prior introduces couplings between $\boldsymbol{\Theta}_i^{(c)}$ for all conditions c which have the same causal mechanism (i.e., if $f_i^{(c')} = f_i^{(c)}$). An intervention may change f_i into another function $\tilde{f}_i$; whether or not such a mechanism change happens, depends on the experimental condition c and on the hypothetical causal structure $\mathcal{G}$ (remember that an abundance intervention on compound i changes its causal mechanism f_i and an activity intervention on compound i changes the causal mechanisms of its children $\{f_j\}_{j \in \mathrm{ch}_{\mathcal{G}}(i)}$). Let the causal mechanism for compound i in condition c be given by $f_i^{(c)} = \phi_{i, m_{ic}(\mathcal{G})}$, where the label $m_{ic}(\mathcal{G}) \in \{1, 2, \dots, M_i(\mathcal{G})\}$ depends on the causal structure $\mathcal{G}$ (see also Figure 1). Here, $M_i(\mathcal{G})$ is the total number of different causal mechanisms for compound i needed to account for all experimental conditions. We take a prior that couples parameters corresponding to the same causal mechanisms:

$$p(\boldsymbol{\Theta} \,|\, \mathcal{G}) = \prod_{i=1}^{D} \prod_{m=1}^{M_i(\mathcal{G})} p\big((\boldsymbol{\Theta}_i^{(c)})_{m_{ic}(\mathcal{G})=m} \,|\, \mathcal{G}\big).$$

Note that this prior couples parameters $\mathbf{\Theta}_i^{(c)}$ with $\mathbf{\Theta}_i^{(c')}$ only if $m_{ic}(\mathcal{G}) = m_{ic'}(\mathcal{G})$.

We will consider two different choices for the factors $p\big((\mathbf{\Theta}_i^{(c)})_{m_{ic}(\mathcal{G})=m} \,|\, \mathcal{G}\big)$, corresponding to different degrees of approximation of the fact that the parameters $\{\mathbf{\Theta}_i^{(c)}\}_{c=1}^K$ correspond with linearizations of the latent nonlinear causal mechanisms $\{\phi_{i,m}\}_{m=1}^{M_i(\mathcal{G})}$.

2.5.1 Linear mechanisms prior

This prior assumes that no relinearizations of the descendants of an intervention node are required. In other words, if one or more causal mechanisms change as a result of some intervention, the input distributions of the descendant variables are assumed to change not too much, such that their linearization remains approximately the same. We can then use hard equality constraints:

$$p\big((\mathbf{\Theta}_i^{(c)})_{m_{ic}(\mathcal{G})=m} \,|\, \mathcal{G}\big) = \int p(\mathbf{\Theta}_i^m) \prod_{\substack{c=1 \\ m_{ic}(\mathcal{G})=m}}^{K} \delta(\mathbf{\Theta}_i^{(c)} - \mathbf{\Theta}_i^m)\, d\mathbf{\Theta}_i^m$$

with

$$p\big(\mathbf{\Theta}_i^m = (\boldsymbol{b}, \mu, a)\big) = \mathcal{N}(\boldsymbol{b} \,|\, \mathbf{0}_D, \lambda^2 \mathrm{diag}(\mathcal{G}_{i,\cdot}))\, \mathcal{N}(\mu \,|\, 0, \tau)\, \mathcal{N}(a \,|\, 0, \tau)$$

where $a = \log \alpha$ and where we let $\tau \to \infty$, which yields a flat prior over the location μ_i and Jeffrey's prior over the scale α_i. We have a single hyperparameter λ for penalizing the nonzero components of $\boldsymbol{b} = \boldsymbol{B}_{\cdot i}^{(c)}$.

2.5.2 Nonlinear mechanisms prior

The previous prior does not deal well with the situation in Figure 2(a). Here, condition A could be the baseline (observational condition), and condition B could be an intervention that changes something upstream of $\boldsymbol{x}_i$, but keeps the mechanism f_i unchanged. Because the upstream intervention may lead to a change in input distribution of the parents $\boldsymbol{x}_{\mathrm{pa}(i)}$, relinearization of f_i around a new average input is desirable in general. Therefore, we introduce a prior that allows for downstream relinearizations. We have tried a prior that allows *all* descendants of an intervention target in condition $c \neq 1$ to pick parameters $\mathbf{\Theta}_i^{(c)}$ *independent* of the baseline parameters $\mathbf{\Theta}_i^{(1)}$ in the observational setting $c = 1$. That prior does yield better results in the acyclic case than the prior in Section 2.5.1, but in the cyclic case it leads to "cheating" in the sense that the prior strongly encourages to introduce one big directed cycle that connects all the variables. Then, each variable is a descendant of each other variable, and can pick new (independent) parameters in *each* experimental condition, effectively completely decoupling all experimental conditions.

The solution we propose here is a compromise that replaces the hard equality constraints of the prior in section 2.5.1 by soft constraints. The idea is to model each causal mechanism $f_i^{m_{ic}(\mathcal{G})}(\boldsymbol{x}_{\mathrm{pa}(i)}, \epsilon_i)$ as a Gaussian Process (GP) and interpret the parameters $(B_{\cdot i}^{(c)}, \mu_i^{(c)}, \alpha_i^{(c)})$ as pseudo-data for the GP (Solak et al., 2003). They note that for Gaussian Process regression, one is not necessarily restricted to using pairs of input and output, but one can combine this data with data regarding the derivative of the output with respect to some input dimension, at a given input location. In our case, the "data" are actually the linearized parameters $(B_{\cdot i}^{(c)}, \mu_i^{(c)}, \alpha_i^{(c)})$, which are coupled to the real data via the likelihood (4). We use an isotropic squared exponential covariance function:

$$k\big((\boldsymbol{x}_{\mathrm{pa}(i)}, \epsilon_i), (\tilde{\boldsymbol{x}}_{\mathrm{pa}(i)}, \tilde{\epsilon}_i)\big) = \sigma_{\mathrm{out}}^2 \exp\left(-\frac{(\boldsymbol{x}_{\mathrm{pa}(i)} - \tilde{\boldsymbol{x}}_{\mathrm{pa}(i)})^2}{2\sigma_{\mathrm{in}}^2}\right) \exp\left(-\frac{(\epsilon_i - \tilde{\epsilon}_i)^2}{2\sigma_{\mathrm{in}}^2}\right)$$

and add a small "jitter" term for numerical stability purposes (i.e., we add $\sigma_{\mathrm{jitter}}^2 \boldsymbol{I}$ to the kernel matrix $\boldsymbol{K}$). Similar to the prior in 2.5.1, this GP prior couples different c for the same i. As the determinant factor in the likelihood couples different i for the same c, we cannot simply use the trick of Solak et al. (2003) (who use the posterior distribution of the biases and slopes of Bayesian linear regressions as pseudo-data), but have to apply a more global approximation scheme (see Section 2.6).

This prior deals well with the situation in Figure 2(a), as the pseudo-data corresponding to the two local linear models would have a high probability under this GP prior. On the other hand, the GP prior strongly penalizes situations such as in Figure 2(b), in line with our intuition that the same causal mechanism f_i cannot be a good model for the data of both condition A and B in that case.

2.6 Structure priors and scoring structures

We use an approximate Bayesian approach to calculate the posterior probability of a putative causal graph $\mathcal{G}$, given the data and prior assumptions. In principle, exact Bayesian scoring would yield automatic regularization (if our assumption that there is no confounding holds true). However, as the posterior distribution is intractable, we have to approximate it. Given a hypothetical causal structure $\mathcal{G}$, we numerically optimize the posterior with respect to the parameter and employ the Laplace approximation (Laplace, 1774) to get an approximation of the evidence (marginal likelihood) for that structure.

The number of possible causal graphs $\mathcal{G}$ grows very quickly as a function of the number of variables: for the Sachs et al. (2005) data, which has $D = 11$ variables, there are about 3.1×10^{22} different directed acyclic graphs (DAGs) and $2^{D^2-D} \approx 1.2 \times 10^{33}$ directed graphs. Even though calculating the evidence for a single structure is doable, exhaustive enumeration or scoring is clearly hopeless. Therefore, we use greedy optimization methods (local search) in the hope to find the important modes of the posterior over causal structures. We use simple priors over structures: a flat prior over directed graphs, a flat prior over acyclic graphs, and flat priors over all graphs (either acyclic or all directed graphs) that have at most n edges.

If exact Bayesian inference were feasible, we could either select the best scoring structure, or average over structures according to their evidence, in order to obtain predictions. However, as we are using approximate inference, we will also use *stability selection* (Meinshausen and Bühlmann, 2010) to assess the stability of posterior edge probabilities.

3 Application on real-world data

In this section, we describe the results of our proposed method on the flow cytometry data set.

3.1 Properties of the data

The data published by Sachs et al. (2005) is a good test case for causal discovery methods for several reasons. First, the high quality of the data:[3] each sample is a multivariate measurement in a *single cell* (usually, only population averages are measured), the number of data points is large (about 10^4 in total), and the measurement noise seems to be relatively low. Furthermore, knowledge about the "ground truth" is available, which helps verification of results. Finally, good results have already been demonstrated with acyclic causal discovery methods, but the data is interesting for our purposes as it shows evidence of feedback relationships.

Figure 3(a) shows a subset of the data as a heat map. Table 1 describes the biological background knowledge about the different experimental conditions: which reagent has been added, and what is the known effect of this reagent? We used a subset of 8 of the available 14 experimental conditions. Figure 3(b) shows whether the interventional distributions are significantly different from the observational distribution, for each variable and each experimental condition. Figure 4 shows two scatter plots of the data in two different experimental conditions. Note the almost perfect linear relationship between log-abundance of Raf and Mek in condition 5, which implies that the measurement noise (i.e., the noise added by the measurement device) must be relatively small. This also shows a strong dependence between Raf and Mek, which is expected from the consensus network (where Raf is a direct cause of Mek). On the other hand, note the absence of dependence between Mek and Erk. Assuming the consensus (Mek causes Erk) to be true, this is an example of a *faithfulness* violation. The data actually shows more such faithfulness violations, which makes causal discovery challenging (but not necessarily impossible, since we do have interventional data). Furthermore, note that the intervention on Mek (condition 5) changes the Raf concentration. So, assuming that the consensus that Raf causes Mek is true, this is an example of feedback. Another example of feedback is that changing the activity of Mek results in a change of abundance of Mek itself.[4] Finally, the figure shows one more aspect of the data: it is censored by the detection limit of the measurement device (i.e., all abundances lower than some threshold $\theta = 1$ are assigned the value θ).

Table 1: Experimental metadata: conditions we used for inferring the causal structure. The information about the type of intervention is used as background knowledge for causal discovery.

c	Reagent	Intervention
1	-	none (observational)
2	Akt-inhibitor	inhibits AKT activity
3	G0076	inhibits PKC activity
4	Psitectorigenin	inhibits PIP2 abundance
5	U0126	inhibits MEK activity
6	LY294002	changes PIP2/PIP3 mechanisms
7	PMA	activates PKC activity
8	β2CAMP	activates PKA activity

3.2 Results

The consensus network and the reconstruction by Sachs et al. (2005) are illustrated in Figure 5.

We experimented with several different combinations of structure and parameter priors. We used hyperparameter $\lambda = 10$ for the linear mechanisms prior (Section 2.5.1), and $\sigma_{\text{in}} = \sigma_{\text{out}} = 10$ for the nonlinear

[3]However, we did discover an error in the published data: the first 848 measurements of RAF and MEK in the third experimental condition (AKT-inhibitor) are identical to those in the seventh condition (LY294002). We informed the authors about this and decided to ignore this issue here.

[4]An alternative explanation of these observations could be non-specificity of the intervention reagents (Eaton and Murphy, 2007).

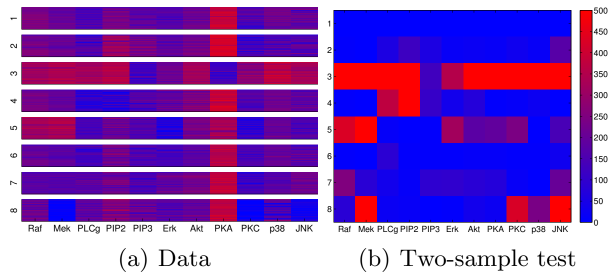


Figure 3: (a) Subset of the data from Sachs et al. (2005). Color corresponds with log-abundance (red is high, blue is low); columns correspond with compounds (phosphorylated proteins and phospholipids); numbered subsets correspond with different experimental conditions (see also Table 1); lines within a row correspond with data samples (i.e., individual cells). (b) Negative log p-value of the Kolmogorov-Smirnov two-sample test, comparing the data of condition c (on the vertical axis) with the observational data (condition $c = 1$). Color indicates how significantly different the two distributions are (red meaning that the difference is extremely significant).

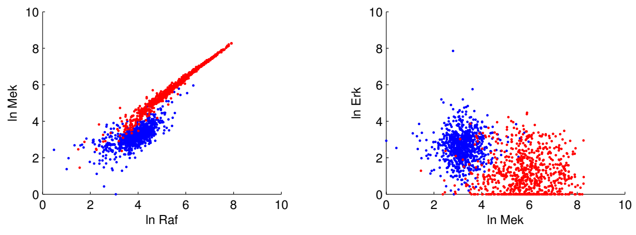


Figure 4: Scatter plot of log abundances of Mek vs. Raf (left) and Erk vs. Mek (right). Blue: condition 1 (no intervention); Red: condition 5 (MEK inhibitor).

mechanisms prior (Section 2.5.2), with $\sigma_{\text{jitter}} = 0.01$. Using smaller values of the jitter did not yield significantly different results, but increased computation time considerably. Figure 6 shows how the log-evidence depends on n, the maximum number of edges. Each point in the plot is the result of a new greedy optimization from a different random starting point. Especially for higher numbers of edges, local maxima over structures are present, but we often seem to find the global maximum with only a few restarts of the local search procedure. Our stability selection results with a constraint on the maximum number of edges are shown in Figure 5(c) (with acyclicity constraint) and Figure 7 (cycles allowed).

In the strongly regularized acyclic case (Figure 5(c)) the precise form of the multitask prior is not very relevant: almost identical results are obtained with the (non)linear prior and/or (non-)Gaussian noise (not shown). The selected edges are very robust. Notice

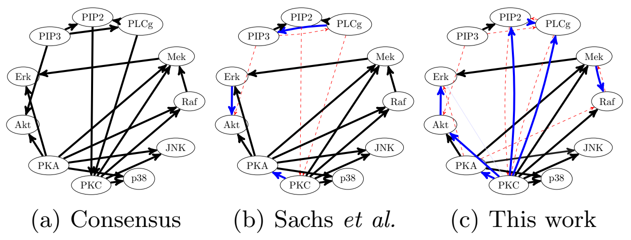


Figure 5: (a) Consensus network, according to Sachs et al. (2005); (b) Reconstruction of the signaling network by Sachs et al. (2005), in comparison with the consensus network; (c) Our best *acyclic* reconstruction with at most 17 edges. Black edges: expected. Blue edges: unexpected, novel findings. Red dashed edges: missing.

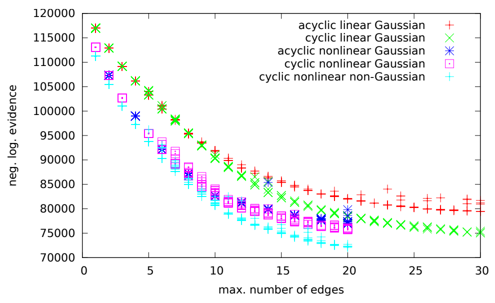


Figure 6: Negative log-evidence as a function of the maximum number of edges. Each point is a local optimum with respect to structures.

that our reconstruction shows less similarity with the consensus network than the reconstruction of Sachs et al. (2005) (cf. Figure 5(b)). However, when looking more closely at the unexpected edges in our acyclic reconstruction, one sees that they actually explain the data quite well. For example, our finding that Mek causes Raf (instead of vice versa) is consistent with the strong change in Raf abundance due to the Mek inhibitor (condition 5, see also Figure 4 and Figure 3(b)).[5] Similarly, the other unexpected edges in our reconstruction can all be understood qualitatively by combining the information in Figure 3(b) with that in Table 1.

When allowing for cycles, the dependence on the prior is more noticeable (see Figure 7). Nevertheless, there is reasonable agreement between the results for different priors. We see evidence for three two-cycles: Mek $\rightleftarrows$ PKC, Akt $\rightleftarrows$ Erk and Mek $\rightleftarrows$ PKA. When regularizing less strongly by increasing n (the number of edges), re-

[5] Given this strong effect, it is surprising that Sachs et al. (2005) do find the opposite arrow. Presumably this is due to the fact that they are using information about the sign of the activity intervention (i.e., whether it is an activator or an inhibitor).

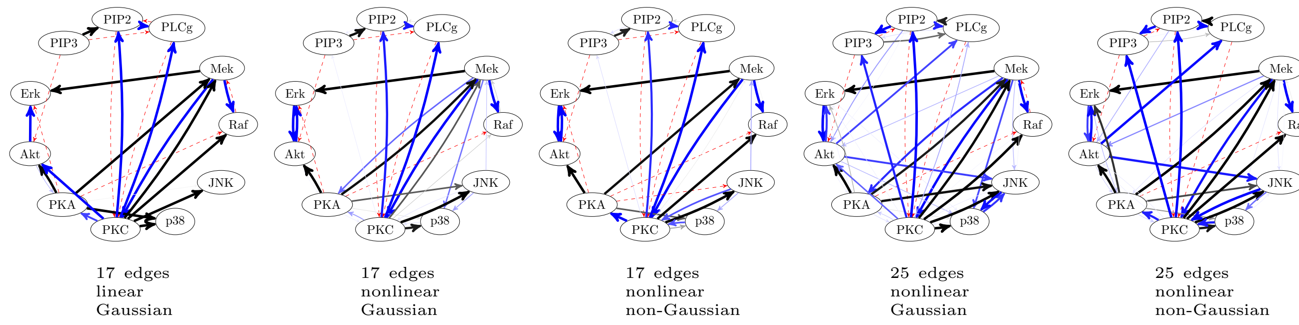


Figure 7: Stability selection results with a constraint on the number of edges, for various priors. Edge thickness and intensity reflect the probability of selecting that edge in the stability selection procedure.

Table 2: Negative log-evidences of our estimated structures (with max. 17 edges) for various structure and parameter priors in comparison with the negative log-evidences of the consensus structure and optimal structure found by Sachs et al. (2005) with the same parameter prior. All values are in units of 10^3.

Structure & Parameter Prior	Consensus	Sachs	This work
Acyclic, linear, Gaussian	96.5	92.0	**83.7**
Cyclic, linear, Gaussian	96.6	92.1	**80.4**
Acyclic, nonlinear, Gaussian	87.8	81.8	**77.7**
Cyclic, nonlinear, Gaussian	87.8	81.8	**76.6**
Cyclic, nonlinear, non-Gaussian	85.4	79.2	**72.9**

sults become more prior dependent. There also seems to be some evidence for a two-cycle PIP2 $\rightleftarrows$ PLCg.

In the acyclic case, parameter estimates conditional on graph structure are very robust. In the cyclic case, this no longer holds, and parameters can often not be estimated reliably from the data (as can be concluded from their posterior variance according to the Laplace approximation, but also from the lack of robustness of their estimates and the occurence of local maxima of the posterior parameter distribution). Empirically, we observed that the *structure* of the estimated graph is much more robust, though.

In Table 2 we compare the scores of some of our structures with the score of the consensus structure and that of the reconstruction by Sachs et al. (2005). Unsurprisingly, our scores are always at least as good (because they result from an optimization of scores over structures, whereas the other structures are fixed), but in all cases, the improvement is considerable.

4 Discussion

Performing a proper causal analysis of the (Sachs et al., 2005) data is a challenging task for various reasons. First of all, time-series data are absent, so we can only work under the equilibrium assumption. Both confounders and feedback loops are expected to be present. Most of the interventions cannot be appropriately modeled with the standard formalism, the "do-operator" (Pearl, 2000), but need to be modeled in another way. Furthermore, assumptions about the specificity of interventions may be unrealistic. Finally, several strong faithfulness violations seem to be present. This work addresses several of these issues.

Our analysis confirms the hypothesis that several feedback loops are present in the underlying system. We showed that our method gives a more accurate quantitative description of the data at comparable model complexity compared to existing methods. An interesting question from the causal point of view is whether or not our method also gives more accurate predictions for the effects of unseen interventions. We hope to address this question in the future. However, it is likely that it can only be answered definately by carrying out additional validation experiments.

We observed empirically that in the cyclic case, the parameters are often not identifiable, even though the structure is. This observation has important implications for the ability to make predictions for unseen interventions: even though reliable qualitative predictions seem possible (e.g., "an intervention on x_i has (no) effect on x_j"), quantitative predictions depend strongly on the parameter estimates. As the parameters cannot be estimated reliably from this data, the quantitative predictions will be unreliable as well. This does not mean that making such quantitative predictions is hopeless in principle, though. Indeed, the alternative conclusion could simply be that more experimental data is needed in order to do so reliably.

In future work, we plan to compare our local linearization approach with other approximations, e.g., FITC (Snelson and Ghahramani, 2006). Also, a way to take into account the information about the sign of the activity intervention may further improve the results. Finally, we hope to find collaborators for experimental validation of our findings.

Acknowledgements

We thank Bram Thijssen, Tjeerd Dijkstra and Tom Claassen for stimulating discussions. We also thank Karen Sachs for kindly answering our questions about the data. JM was supported by NWO, the Netherlands Organization for Scientific Research (VENI grant 639.031.036).

References

Bollen, K. A. (1989). *Structural Equations with Latent Variables*. John Wiley & Sons.

Eaton, D. and Murphy, K. (2007). Exact bayesian structure learning from uncertain interventions. In *Proceedings AISTATS 2007*.

Eberhardt, F., Hoyer, P. O., and Scheines, R. (2010). Combining experiments to discover linear cyclic models with latent variables. *Proceedings AISTATS 2010*, pages 185–192.

Hoyer, P. O., Janzing, D., Mooij, J. M., Peters, J., and Schölkopf, B. (2009). Nonlinear causal discovery with additive noise models. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 21 (NIPS*2008)*, pages 689–696.

Hyttinen, A., Eberhardt, F., and Hoyer, P. (2012). Learning linear cyclic causal models with latent variables. *Journal for Machine Learning Research*, 13:33873439.

Itani, S., Ohannessian, M., Sachs, K., Nolan, G. P., and Dahleh, M. A. (2010). Structure learning in causal cyclic networks. In *JMLR Workshop and Conference Proceedings*, volume 6, page 165176.

Lacerda, G., Spirtes, P., Ramsey, J., and Hoyer, P. O. (2008). Discovering cyclic causal models by independent components analysis. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-2008)*.

Laplace, P. (1774). Memoir on the probability of causes of events. In *Mémoires de Mathématique et de Physique, Tome Sixième*.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection (with discussion). *Journal of the Royal Statistical Society Series B*, 72:417–473.

Mooij, J. M., Janzing, D., Heskes, T., and Schölkopf, B. (2011). On causal discovery with cyclic additive noise models. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 24 (NIPS*2011)*, pages 639–647.

Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press.

Peters, J., Mooij, J. M., Janzing, D., and Schölkopf, B. (2011). Identifiability of causal graphs using functional models. In Cozman, F. G. and Pfeffer, A., editors, *Proceedings of the 27th Annual Conference on Uncertainty in Artificial Intelligence (UAI-11)*, pages 589–598. AUAI Press.

Rasmussen, C. E. and Williams, C. (2006). *Gaussian Processes for Machine Learning*. MIT Press.

Sachs, K., Perez, O., Pe'er, D., Lauffenburger, D., and Nolan, G. (2005). Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308:523–529.

Schmidt, M. and Murphy, K. (2009). Modeling discrete interventional data using directed cyclic graphical models. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI-09)*.

Shimizu, S., Hoyer, P. O., Hyvärinen, A., and Kerminen, A. J. (2006). A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030.

Snelson, E. and Ghahramani, Z. (2006). Sparse gaussian processes using pseudo-inputs. In Weiss, Y., Schölkopf, B., and Platt, J., editors, *Advances in Neural Information Processing Systems 18 (NIPS*2005)*, pages 1257–1264. MIT Press, Cambridge, MA.

Solak, E., Murray-Smith, R., Leithead, W. E., Leith, D. J., and Rasmussen, C. E. (2003). Derivative observations in gaussian process models of dynamic systems. In S. Becker, S. T. and Obermayer, K., editors, *Advances in Neural Information Processing Systems 15 (NIPS*2002)*, pages 1033–1040. MIT Press, Cambridge, MA.

Zhang, K. and Hyvärinen, A. (2009). On the identifiability of the post-nonlinear causal model. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI-09)*.

From Ordinary Differential Equations to Structural Causal Models: the deterministic case

Joris M. Mooij[*]
Institute for Computing and Information Sciences
Radboud University Nijmegen
The Netherlands

Dominik Janzing
Max Planck Institute
for Intelligent Systems
Tübingen, Germany

Bernhard Schölkopf
Max Planck Institute
for Intelligent Systems
Tübingen, Germany

Abstract

We show how, and under which conditions, the equilibrium states of a first-order Ordinary Differential Equation (ODE) system can be described with a deterministic Structural Causal Model (SCM). Our exposition sheds more light on the concept of causality as expressed within the framework of Structural Causal Models, especially for cyclic models.

1 Introduction

Over the last few decades, a comprehensive theory for acyclic causal models was developed (e.g., see (Pearl, 2000; Spirtes et al., 1993)). In particular, different, but related, approaches to causal inference and modeling have been proposed for the causally sufficient case. These approaches are based on different starting points. One approach starts from the (local or global) causal Markov condition and links observed independences to the causal graph. Another approach uses causal Bayesian networks to link a particular factorization of the joint distribution of the variables to causal semantics. The third approach uses a structural causal model (sometimes also called structural equation model or functional causal model) where each effect is expressed as a function of its direct causes and an unobserved noise variable. The relationships between these aproaches are well understood (Lauritzen, 1996; Pearl, 2000).

Over the years, several attempts have been made to extend the theory to the cyclic case, thereby enabling causal modeling of systems that involve feedback (Spirtes, 1995; Koster, 1996; Pearl and Dechter, 1996; Neal, 2000; Hyttinen et al., 2012). However, the relationships between the different approaches mentioned before do not immediately generalize to the cyclic case in general (although partial results are known for the linear case and the discrete case). Nevertheless, several algorithms (starting from different assumptions) have been proposed for inferring cyclic causal models from observational data (Richardson, 1996; Lacerda et al., 2008; Schmidt and Murphy, 2009; Itani et al., 2010; Mooij et al., 2011).

The most straightforward extension to the cyclic case seems to be offered by the structural causal model framework. Indeed, the formalism stays intact when one simply drops the acyclicity constraint. However, the question then arises how to interpret cyclic structural equations. One option is to assume an underlying discrete-time dynamical system, in which the structural equations are used as fixed point equations (Spirtes, 1995; Dash, 2005; Lacerda et al., 2008; Mooij et al., 2011; Hyttinen et al., 2012), i.e., they are used as update rules to calculate the values at time $t+1$ from the values at time t, and then one lets $t \to \infty$. Here we show how an alternative interpretation of structural causal models arises naturally when considering systems of ordinary differential equations. By considering how these differential equations behave in an equilibrium state, we arrive at a structural causal model that is time independent, yet where the causal semantics pertaining to interventions is still valid. As opposed to the usual interpretation as discrete-time fixed point equations, the continuous-time dynamics is not defined by the structural equations. Instead, we describe how the structural equations arise from the given dynamics. Thus it becomes evident that different dynamics can yield identical structural causal models. This interpretation sheds more light on the meaning of structural equations, and does not make any substantial distinction between the cyclic and acyclic cases.

It is sometimes argued that inferring causality amounts to simply inferring the time structure connecting the observed variables, since the cause always preceeds the effect. This, however, ignores two important facts: First, time order between two variables

[*]Also affiliated to the Informatics Institute, University of Amsterdam, The Netherlands

does not tell us whether the earlier one caused the later one, or whether both are due to a common cause. This paper addresses a second counter argument: a variable need not necessarily refer to a measurement performed at a certain time instance. Instead, a causal graph may formalize how intervening on some variables influences the equilibrium state of others. This describes a phenomenological level on which the original time structure between variables disappears, but causal graphs und structural equations may still be well-defined. On this level, also cyclic structural equations get a natural and well-defined meaning.

For simplicity, we consider only deterministic systems, and leave the extension to stochastic systems with possible confounding as future work.

2 Ordinary Differential Equations

Let $\mathcal{I} := \{1, \dots, D\}$ be an index set of variable labels. Consider variables $X_i \in \mathcal{R}_i$ for $i \in \mathcal{I}$, where $\mathcal{R}_i \subseteq \mathbb{R}^{d_i}$. We use normal font for a single variable and boldface for a tuple of variables $\boldsymbol{X}_I \in \prod_{i \in I} \mathcal{R}_i$ with $I \subseteq \mathcal{I}$.

2.1 Observational system

Consider a dynamical system $\mathcal{D}$ described by D coupled first-order ordinary differential equations and an initial condition $\boldsymbol{X}_0 \in \mathcal{R}_{\mathcal{I}}$:[1]

$$\dot{X}_i(t) = f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)}), \quad X_i(0) = (\boldsymbol{X}_0)_i \quad \forall i \in \mathcal{I} \tag{1}$$

Here, $\mathrm{pa}_{\mathcal{D}}(i) \subseteq \mathcal{I}$ is the set of (indices of) *parents*[2] of variable X_i, and each $f_i : \mathcal{R}_{\mathrm{pa}_{\mathcal{D}}(i)} \to \mathcal{R}_i$ is a (sufficiently smooth) function. This dynamical system is assumed to describe the "natural" or "observational" state of the system, without any intervention from outside. We will assume that if $j \in \mathrm{pa}_{\mathcal{D}}(i)$, then f_i depends on X_j (in other words, f_i should not be constant when varying X_j). Slightly abusing terminology, we will henceforth call such a dynamical system $\mathcal{D}$ an Ordinary Differential Equation (ODE).

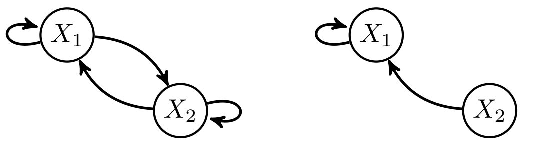
The *structure* of these differential equations can be represented as a directed graph $\mathcal{G}_{\mathcal{D}}$, with one node for each variable and a directed edge from X_i to X_j if and only if $\dot{X}_j$ depends on X_i.

2.1.1 Example: the Lotka-Volterra model

The Lotka-Volterra model (Murray, 2002) is a well-known model from population biology, modeling the mutual influence of the abundance of prey $X_1 \in [0, \infty)$ (e.g., rabbits) and the abundance of predators $X_2 \in$

[1]We write $\dot{X} := \frac{dX}{dt}$.

[2]Note that X_i can be a parent of itself.



(a) $\mathcal{G}_{\mathcal{D}}$ (b) $\mathcal{G}_{\mathcal{D}_{\mathrm{do}(X_2=\xi_2)}}$

Figure 1: (a) Graph of the Lotka-Volterra model (2); (b) Graph of the same ODE after the intervention $\mathrm{do}(X_2 = \xi_2)$, corresponding with (5).

$[0, \infty)$ (e.g., wolves):

$$\begin{cases} \dot{X}_1 &= X_1(\theta_{11} - \theta_{12} X_2) \\ \dot{X}_2 &= -X_2(\theta_{22} - \theta_{21} X_1) \end{cases} \qquad \begin{cases} X_1(0) = a \\ X_2(0) = b \end{cases} \tag{2}$$

with all parameters $\theta_{ij} > 0$ and initial condition satisfying $a \geq 0, b \geq 0$. The graph of this system is depicted in Figure 1(a).

2.2 Intervened system

Interventions on the system $\mathcal{D}$ described in (1) can be modeled in different ways. Here we will focus on *"perfect" interventions*: for a subset $I \subseteq \mathcal{I}$ of components, we force the value of $\boldsymbol{X}_I$ to attain some value $\boldsymbol{\xi}_I \in \mathcal{R}_I$. In particular, we will assume that the intervention is active from $t = 0$ to $t = \infty$, and that its value $\boldsymbol{\xi}_I$ does not change over time. Inspired by the do-operator introduced by Pearl (2000), we will denote this type of intervention as $\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)$.

On the level of the ODE, there are many ways of realizing a given perfect intervention. One possible way is to add terms of the form $\kappa(\xi_i - X_i)$ (with $\kappa > 0$) to the expression for $\dot{X}_i$, for all $i \in I$:

$$\dot{X}_i(t) = \begin{cases} f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)}) + \kappa(\xi_i - X_i) & i \in I \\ f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)}) & i \in \mathcal{I} \setminus I, \end{cases} \tag{3}$$
$$X_i(0) = (\boldsymbol{X}_0)_i$$

This would correspond to extending the system by components that monitor the values of $\{X_i\}_{i \in I}$ and exert negative feedback if they deviate from their target values $\{\xi_i\}_{i \in I}$. Subsequently, we let $\kappa \to \infty$ to consider the idealized situation in which the intervention completely overrides the other mechanisms that normally determine the value of $\boldsymbol{X}_I$. Under suitable regularity conditions, we can let $\kappa \to \infty$ and obtain the *intervened system* $\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)}$:

$$\dot{X}_i(t) = \begin{cases} 0 & i \in I \\ f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)}) & i \in \mathcal{I} \setminus I, \end{cases}$$
$$X_i(0) = \begin{cases} \xi_i & i \in I \\ (\boldsymbol{X}_0)_i & i \in \mathcal{I} \setminus I \end{cases} \tag{4}$$

A perfect intervention changes the graph $\mathcal{G}_\mathcal{D}$ associated to the ODE $\mathcal{D}$ by removing the incoming arrows on the nodes corresponding to the intervened variables $\{X_i\}_{i\in I}$. It also changes the parent sets of intervened variables: for each $i \in I$, $\text{pa}_\mathcal{D}(i)$ is replaced by $\text{pa}_{\mathcal{D}_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}}(i) = \emptyset$.

2.2.1 Example: Lotka-Volterra model

Let us return to the example in section 2.1.1. In this context, consider the perfect intervention $\text{do}(X_2 = \xi_2)$. This intervention could be realized by monitoring the abundance of wolves very precisely and making sure that the number equals the target value ξ_2 at all time (for example, by killing an excess of wolves and introducing new wolves from some reservoir of wolves). This leads to the following intervened ODE:

$$\begin{cases}\dot{X}_1 &= X_1(\theta_{11} - \theta_{12}X_2)\\ \dot{X}_2 &= 0\end{cases} \qquad \begin{cases}X_1(0) = a\\ X_2(0) = \xi_2\end{cases} \tag{5}$$

The corresponding intervened graph is illustrated in Figure 1(b).

2.3 Stability

An important concept in our context is *stability*, defined as follows:

Definition 1 *The ODE $\mathcal{D}$ specified in (1) is called* stable *if there exists a unique equilibrium state $\boldsymbol{X}^* \in \mathcal{R}_\mathcal{I}$ such that for any initial state $\boldsymbol{X}_0 \in \mathcal{R}_\mathcal{I}$, the system converges to this equilibrium state as $t \to \infty$:*

$$\exists!_{\boldsymbol{X}^*\in\mathcal{R}_\mathcal{I}}\ \forall_{\boldsymbol{X}_0\in\mathcal{R}_\mathcal{I}} : \lim_{t\to\infty} \boldsymbol{X}(t) = \boldsymbol{X}^*.$$

One can weaken the stability condition by demanding convergence to and uniqueness of the equilibrium only for a certain subset of all initial states. For clarity of exposition, we will use this strong stability condition.

We can extend this concept of stability by considering a certain set of perfect interventions:

Definition 2 *Let $\mathcal{J} \subseteq \mathcal{P}(\mathcal{I})$.*[3] *The ODE $\mathcal{D}$ specified in (1) is called* stable with respect to $\mathcal{J}$ *if for all $I \in \mathcal{J}$ and for all $\boldsymbol{\xi}_I \in \mathcal{R}_I$, the intervened ODE $\mathcal{D}_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ has a unique equilibrium state $\boldsymbol{X}^*_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)} \in \mathcal{R}_\mathcal{I}$ such that for any initial state $\boldsymbol{X}_0 \in \mathcal{R}_\mathcal{I}$ with $(\boldsymbol{X}_0)_I = \boldsymbol{\xi}_I$, the system converges to this equilibrium as $t \to \infty$:*

$$\exists!_{\boldsymbol{X}^*_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}\in\mathcal{R}_\mathcal{I}}\ \forall_{\substack{\boldsymbol{X}_0\in\mathcal{R}_\mathcal{I} s.t.\\ (\boldsymbol{X}_0)_I=\boldsymbol{\xi}_I}} : \lim_{t\to\infty} \boldsymbol{X}(t) = \boldsymbol{X}^*_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}.$$

[3]For a set A, we denote with $\mathcal{P}(A)$ the power set of A (the set of all subsets of A).

This definition can also be weakened by not demanding stability for all $\boldsymbol{\xi}_I \in \mathcal{R}_I$, but for smaller subsets instead. Again, we will use this strong condition for clarity of exposition, although in a concrete example to be discussed later (see Section 2.3.2), we will actually weaken the stability assumption along these lines.

2.3.1 Example: the Lotka-Volterra model

The ODE (2) of the Lotka-Volterra model is not stable, as discussed in detail by Murray (2002). Indeed, it has two equilibrium states, $(X_1^*, X_2^*) = (0, 0)$ and $(X_1^*, X_2^*) = (\theta_{22}/\theta_{21}, \theta_{11}/\theta_{12})$. The Jacobian of the dynamics is given by:

$$\nabla \boldsymbol{f}(\boldsymbol{X}) = \begin{pmatrix}\theta_{11} - \theta_{12}X_2 & -\theta_{12}X_1\\ \theta_{21}X_2 & -\theta_{22} + \theta_{21}X_1\end{pmatrix}$$

In the first equilibrium state, it has a positive and a negative eigenvalue (θ_{11} and $-\theta_{22}$, respectively), and hence this equilibrium is unstable. In the second equilibrium state it has two imaginary eigenvalues, $\pm i\sqrt{\theta_{11}\theta_{22}}$. One can show (Murray, 2002) that the steady state of the system is an undamped oscillation around this equilibrium.

The intervened system (5) is only generically stable, i.e., for most values of ξ_2: the unique stable equilibrium state is $(X_1^*, X_2^*) = (0, \xi_2)$ as long as $\theta_{11} - \theta_{12}\xi_2 \neq 0$. If $\theta_{11} - \theta_{12}\xi_2 = 0$, there exists a family of equilibria $(X_1^*, X_2^*) = (c, \xi_2)$ with $c \geq 0$.

2.3.2 Example: damped harmonic oscillators

The favorite toy example of physicists is a system of coupled harmonic oscillators. Consider a one-dimensional system of D point masses m_i ($i = 1, \ldots, D$) with positions $Q_i \in \mathbb{R}$ and momenta $P_i \in \mathbb{R}$, coupled by springs with spring constants k_i and equilibrium lengths l_i, under influence of friction with friction coefficients b_i, with fixed end positions $Q = 0$ and $Q = L$ (see also Figure 2).

We first sketch the qualitative behavior: there is a unique equilibrium position where the sum of forces vanishes for every single mass. Moving one or several masses out of their equilibrium position stimulates vibrations of the entire system. Damped by friction, every mass converges to its unique and stable equilibrium position in the limit of $t \to \infty$. If one or several

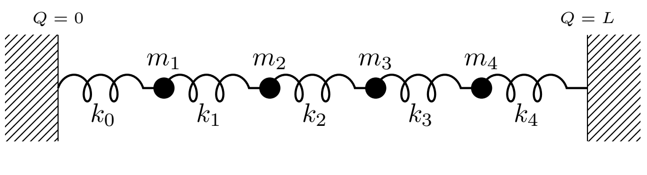
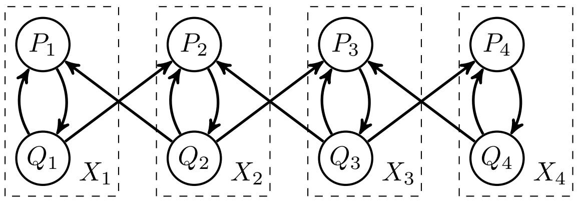
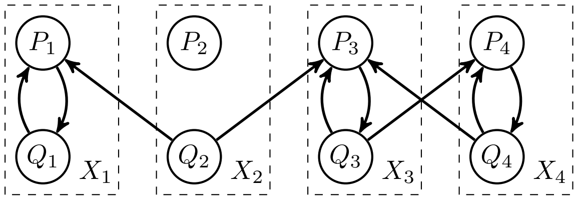


Figure 2: Mass-spring system for $D = 4$.



(a) $\mathcal{G}_{\mathcal{D}}$



(b) $\mathcal{G}_{\mathcal{D}_{\mathrm{do}(Q_2=\xi_2, P_2=0)}}$

Figure 3: Graphs of the dynamics of the mass-spring system for $D = 4$. (a) Observational situation (b) Intervention $\mathrm{do}(Q_2 = \xi_2, P_2 = 0)$.

masses are fixed to positions different from their equilibrium points, the positions of the remaining masses still converge to unique (but different) equilibrium positions. The structural equations that we derive later will describe the change of the unconstrained equilibrium positions caused by fixing the others.

The equations of motion for this system are given by:

$$\begin{aligned}\dot{P}_i &= k_i(Q_{i+1} - Q_i - l_i) \\ &\quad - k_{i-1}(Q_i - Q_{i-1} - l_{i-1}) - \frac{b_i}{m_i} P_i \\ \dot{Q}_i &= P_i / m_i\end{aligned}$$

where we define $Q_0 := 0$ and $Q_{D+1} := L$. The graph of this ODE is depicted in Figure 3(a). At equilibrium (for $t \to \infty$), all momenta vanish, and the following equilibrium equations hold:

$$\begin{aligned}0 &= k_i(Q_{i+1} - Q_i - l_i) - k_{i-1}(Q_i - Q_{i-1} - l_{i-1}) \\ 0 &= P_i\end{aligned}$$

which is a linear system of equations in terms of the Q_i. There are D equations for D unknowns $Q_1, \dots, Q_D$, and one can easily check that it has a unique solution.

A perfect intervention on Q_i corresponds to fixating the position of the i'th mass. Physically, this is achieved by adding a force that drives Q_i to some fixed location, i.e., the intervention on Q_i is achieved through modifying the equation of motion for $\dot{P}_i$. To deal with this example in our framework, we consider the pairs $X_i := (Q_i, P_i) \in \mathbb{R}^2$ to be the elementary variables. Consider for example the perfect intervention $\mathrm{do}(X_2 = (\xi_2, 0))$, which effectively replaces the dynamical equations $\dot{Q}_2$ and $\dot{P}_2$ by $\dot{Q}_2 = 0$, $\dot{P}_2 = 0$ and their initial conditions by $(\boldsymbol{Q}_0)_2 = \xi_2$, $(\boldsymbol{P}_0)_2 = 0$. The graph of the corresponding ODE is depicted in Figure 3(b). Because of the friction, also this intervened system converges to a unique equilibrium that does not depend on the initial value.

This holds more generally: for any perfect intervention on (any number) of pairs X_i of the type $\mathrm{do}(X_i = (\xi_i, 0))$, the intervened system will converge towards a unique equilibrium because of the damping term. Interventions that result in a nonzero value for any momentum P_i while the corresponding position is fixed are physically impossible, and hence will not be considered. Concluding, we have seen that the mass-spring system is stable with respect to perfect interventions on any number of position variables, which we model mathematically as a joint intervention on the corresponding pairs of position and momentum variables.

3 Equilibrium equations

In this section, we will study how the dynamical equations give rise to *equilibrium equations* that describe equilibrium states, and how these change under perfect interventions. This is an intermediate representation on our way to structural causal models.

3.1 Observational system

At equilibrium, the rate of change of any variable is zero, by definition. Therefore, an equilibrium state of the observational system $\mathcal{D}$ defined in (1) satisfies the following *equilibrium equations*:

$$0 = f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)}) \qquad \forall i \in \mathcal{I}. \tag{6}$$

This is a set of D coupled equations with unknowns $X_1, \dots, X_D$. The stability assumption (cf. Definition 1) implies that there exists a unique solution $\boldsymbol{X}^*$ of the equilibrium equations (6).

3.2 Intervened system

Similarly, for the intervened system $\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_i = \boldsymbol{\xi}_i)}$ defined in (4), we obtain the following equilibrium equations:

$$\begin{cases} 0 = X_i - \xi_i & \forall i \in \mathcal{I} \\ 0 = f_j(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(j)}) & \forall j \in \mathcal{I} \setminus I \end{cases} \tag{7}$$

If the system is stable with respect to this intervention (cf. Definition 2), then there exists a unique solution $\boldsymbol{X}^*_{\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)}$ of the intervened equilibrium equations (7).

Note that we can also go directly from the equilibrium equations (6) of the observational system $\mathcal{D}$ to the equilibrium equations (7) of the intervened system

$\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$, simply by replacing the equilibrium equations "$0 = f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)})$" for $i \in I$ by equations of the form "$0 = X_i - \xi_i$". Indeed, note that the modified dynamical equation

$$\dot{X}_i = f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)}) + \kappa(\xi_i - X_i)$$

yields an equilibrium equation of the form

$$0 = f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)}) + \kappa(\xi_i - X_i)$$

which, in the limit $\kappa \to \infty$, reduces to $0 = X_i - \xi_i$. This seemingly trivial observation will turn out to be quite important.

3.3 Labeling equilibrium equations

If we would consider the equilibrium equations as a set of *unlabeled* equations $\{\mathcal{E}_i : i \in \mathcal{I}\}$, where $\mathcal{E}_i$ denotes the equilibrium equation "$0 = f_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{D}}(i)})$" (or "$0 = X_i - \xi_i$" after an intervention) for $i \in \mathcal{I}$, then we will not be able to correctly predict the result of interventions, as we do not know *which* of the equilibrium equations should be changed in order to model the particular intervention. This information is present in the dynamical system $\mathcal{D}$ (indeed, the terms "$\dot{X}_i$" in the l.h.s. of the dynamical equations in (1) indicate the targets of the intervention), but is lost when considering the corresponding equilibrium equations (6) as an unlabeled set (because the terms "$\dot{X}_i$" have all been replaced by zeroes).

This important information can be preserved by labeling the equilibrium equations. Indeed, the *labeled* set of equilibrium equations $\mathcal{E} := \{(i, \mathcal{E}_i) : i \in \mathcal{I}\}$ contains all information needed to predict how equilibrium states change on arbitrary (perfect) interventions. Under an intervention $\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)$, the equilibrium equations are changed as follows: for each intervened component $i \in I$, the equilibrium equation $\mathcal{E}_i$ is replaced by the equation $\tilde{\mathcal{E}}_i$ defined as "$0 = X_i - \xi_i$", whereas the other equilibrium equations $\mathcal{E}_j$ for $j \in \mathcal{I} \setminus I$ do not change. If the dynamical system is stable with respect to this intervention, this modified system of equilibrium equations describes the new equilibrium obtained under the intervention. We conclude that the information about the values of equilibrium states and how these change under perfect interventions is encoded in the labeled equilibrium equations.

3.4 Labeled equilibrium equations

The previous considerations motivate the following formal definition of a system of Labeled Equilibrium Equations (LEE) and its semantics under interventions.

Definition 3 *A system of* Labeled Equilibrium Equations (LEE) *$\mathcal{E}$ for D variables $\{X_i\}_{i\in\mathcal{I}}$ with $\mathcal{I} := \{1, \dots, D\}$ consists of D* labeled equations *of the form*

$$\mathcal{E}_i : \quad 0 = g_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{E}}(i)}), \qquad i \in \mathcal{I}, \tag{8}$$

where $\mathrm{pa}_{\mathcal{E}}(i) \subseteq \mathcal{I}$ is the set of (indices of) parents *of variable X_i, and each $g_i : \mathcal{R}_{\mathrm{pa}_{\mathcal{E}}(i)} \to \mathcal{R}_i$ is a function.*

The *structure* of an LEE $\mathcal{E}$ can be represented as a directed graph $\mathcal{G}_{\mathcal{E}}$, with one node for each variable and a directed edge from X_i to X_j (with $j \neq i$) if and only if $\mathcal{E}_i$ depends on X_j.

A perfect intervention transforms an LEE into another (intervened) LEE:

Definition 4 *Let $I \subseteq \mathcal{I}$ and $\boldsymbol{\xi}_I \in \mathcal{R}_I$. For the perfect intervention $\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)$ that forces the variables $\boldsymbol{X}_I$ to take the value $\boldsymbol{\xi}_I$, the intervened LEE $\mathcal{E}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ is obtained by replacing the labeled equations of the original LEE $\mathcal{E}$ by the following modified labeled equations:*

$$0 = \begin{cases} X_i - \xi_i & i \in I \\ g_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{E}}(i)}) & i \in \mathcal{I} \setminus I. \end{cases} \tag{9}$$

We define the concept of solvability for LEEs that mirrors the definition of stability for ODEs:

Definition 5 *An LEE $\mathcal{E}$ is called* solvable *if there exists a unique solution $\boldsymbol{X}^*$ to the system of (labeled) equations $\{\mathcal{E}_i\}$. An LEE $\mathcal{E}$ is called* solvable with respect to *$\mathcal{J} \subseteq \mathcal{P}(\mathcal{I})$ if for all $I \in \mathcal{J}$ and for all $\boldsymbol{\xi}_I \in \mathcal{R}_I$, the intervened LEE $\mathcal{E}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ is solvable.*

As we saw in the previous section, an ODE induces an LEE in a straightforward way. The graph $\mathcal{G}_{\mathcal{E}_{\mathcal{D}}}$ of the induced LEE $\mathcal{E}_{\mathcal{D}}$ is equal to the graph $\mathcal{G}_{\mathcal{D}}$ of the ODE $\mathcal{D}$. It is immediate that if the ODE $\mathcal{D}$ is stable, then the induced LEE $\mathcal{E}_{\mathcal{D}}$ is solvable. As we saw at the end of Section 3.2, our ways of modeling interventions on ODEs and on LEEs are compatible. We will spell out this important result in detail.

Theorem 1 *Let $\mathcal{D}$ be an ODE, $I \subseteq \mathcal{I}$ and $\boldsymbol{\xi}_I \in \mathcal{R}_I$. (i) Applying the perfect intervention $\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)$ to the induced LEE $\mathcal{E}_{\mathcal{D}}$ gives the same result as constructing the LEE corresponding to the intervened ODE $\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$:*

$$(\mathcal{E}_{\mathcal{D}})_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)} = \mathcal{E}_{\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}}.$$

(ii) Stability of the intervened ODE $\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ implies solvability of the induced intervened LEE $\mathcal{E}_{\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}}$, and the corresponding equilibrium and solution $\boldsymbol{X}^*_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ are identical. □

3.4.1 Example: damped harmonic oscillators

Consider again the example of the damped, coupled harmonic oscillators of section 2.3.2. The labeled equilibrium equations are given explicitly by:

$$\mathcal{E}_i : \qquad \begin{cases} 0 &= k_i(Q_{i+1} - Q_i - l_i) \\ & \quad - k_{i-1}(Q_i - Q_{i-1} - l_{i-1}) \\ 0 &= P_i \end{cases} \tag{10}$$

4 Structural Causal Models

In this section we will show how an LEE representation can be mapped to the more popular representation of Structural Causal Models, also known as Structural Equation Models (Bollen, 1989). We follow the terminology of Pearl (2000), but consider here only the subclass of *deterministic* SCMs.

4.1 Observational system

The following definition is a special case of the more general definition in (Pearl, 2000, Section 1.4.1):

Definition 6 *A* deterministic Structural Causal Model (SCM) *$\mathcal{M}$ on D variables $\{X_i\}_{i\in\mathcal{I}}$ with $\mathcal{I} := \{1,\dots,D\}$ consists of D* structural equations *of the form*

$$X_i = h_i(\boldsymbol{X}_{\mathrm{pa}_\mathcal{M}(i)}), \qquad i \in \mathcal{I}, \tag{11}$$

where $\mathrm{pa}_\mathcal{M}(i) \subseteq \mathcal{I} \setminus \{i\}$ *is the set of (indices of)* parents *of variable X_i, and each $h_i : \mathcal{R}_{\mathrm{pa}_\mathcal{M}(i)} \to \mathcal{R}_i$ is a function.*

Each structural equation contains a function h_i that depends on the components of $\boldsymbol{X}$ in $\mathrm{pa}_\mathcal{M}(i)$. We think of the parents $\mathrm{pa}_\mathcal{M}(i)$ as the *direct causes* of X_i (relative to $\boldsymbol{X}_\mathcal{I}$) and the function h_i as the *causal mechanism* that maps the direct causes to the effect X_i. Note that the l.h.s. of a structural equation by definition contains only X_i, and that the r.h.s. is a function of variables *excluding* X_i itself. In other words, X_i is not considered to be a direct cause of itself. The *structure* of an SCM $\mathcal{M}$ is often represented as a directed graph $\mathcal{G}_\mathcal{M}$, with one node for each variable and a directed edge from X_i to X_j (with $j \neq i$) if and only if h_i depends on X_j. Note that this graph does not contain "self-loops" (edges pointing from a node to itself), by definition.

4.2 Intervened system

A Structural Causal Model $\mathcal{M}$ comes with a specific semantics for modeling perfect interventions (Pearl, 2000):

Definition 7 *Let $I \subseteq \mathcal{I}$ and $\boldsymbol{\xi}_I \in \mathcal{R}_I$. For the perfect intervention $\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)$ that forces the variables $\boldsymbol{X}_I$ to take the value $\boldsymbol{\xi}_I$, the intervened SCM $\mathcal{M}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ is obtained by replacing the structural equations of the original SCM $\mathcal{M}$ by the following modified structural equations:*

$$X_i = \begin{cases} \xi_i & i \in I \\ h_i(\boldsymbol{X}_{\mathrm{pa}_\mathcal{M}(i)}) & i \in \mathcal{I} \setminus I. \end{cases} \tag{12}$$

The reason that the equations in a SCM are called "structural equations" (instead of simply "equations") is that they also contain information for modeling interventions, in a similar way as the labeled equilibrium equations contain this information. In particular, the l.h.s. of the structural equations indicate the targets of an intervention.[4]

4.3 Solvability

Similarly to our definition for LEEs, we define:

Definition 8 *An SCM $\mathcal{M}$ is called* solvable *if there exists a unique solution $\boldsymbol{X}^*$ to the system of structural equations. An SCM $\mathcal{M}$ is called* solvable with respect to *$\mathcal{J} \subseteq \mathcal{P}(\mathcal{I})$ if for all $I \in \mathcal{J}$ and for all $\boldsymbol{\xi}_I \in \mathcal{R}_I$, the intervened SCM $\mathcal{M}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ is solvable.*

Note that each (deterministic) SCM $\mathcal{M}$ with acyclic graph $\mathcal{G}_\mathcal{M}$ is solvable, even with respect to the set of all possible intervention targets, $\mathcal{P}(\mathcal{I})$. This is not necessarily true if directed cycles are present.

4.4 From labeled equilibrium equations to deterministic SCMs

Finally, we will now show that under certain stability assumptions on an ODE $\mathcal{D}$, we can represent the information about (intervened) equilibrium states that is contained in the corresponding set of labeled equilibrium equations $\mathcal{E}_\mathcal{D}$ as an SCM $\mathcal{M}_{\mathcal{E}_\mathcal{D}}$.

First, given an LEE $\mathcal{E}$, we will construct an induced SCM $\mathcal{M}_\mathcal{E}$, provided certain solvability conditions hold:

Definition 9 *If the LEE $\mathcal{E}$ is solvable with respect to $\{\mathcal{I} \setminus \{i\}\}_{i\in\mathcal{I}}$, it is called* structurally solvable.

If the LEE $\mathcal{E}$ is structurally solvable, we can proceed as follows. Let $i \in \mathcal{I}$ and write $I_i := \mathcal{I} \setminus$

[4]In Pearl (2000)'s words: "Mathematically, the distinction between structural and algebraic equations is that the latter are characterized by the set of solutions to the entire system of equations, whereas the former are characterized by the solutions of each individual equation. The implication is that any subset of structural equations is, in itself, a valid model of reality—one that prevails under some set of interventions."

$\{i\}$. We define the induced parent set $\mathrm{pa}_{\mathcal{M}_\mathcal{E}}(i) := \mathrm{pa}_\mathcal{E}(i) \setminus \{i\}$. Assuming structural solvability of $\mathcal{E}$, under the perfect intervention $\mathrm{do}(\boldsymbol{X}_{I_i} = \boldsymbol{\xi}_{I_i})$, there is a unique solution $\boldsymbol{X}^*_{\mathrm{do}(\boldsymbol{X}_{I_i}=\boldsymbol{\xi}_{I_i})}$ to the intervened LEE, for any value of $\boldsymbol{\xi}_{I_i} \in \mathcal{R}_{I_i}$. This defines a function $h_i : \mathcal{R}_{\mathrm{pa}_{\mathcal{M}_\mathcal{E}}(i)} \to \mathcal{R}_i$ given by the i'th component $h_i(\boldsymbol{\xi}_{\mathrm{pa}_{\mathcal{M}_\mathcal{E}}(i)}) := \big(\boldsymbol{X}^*_{\mathrm{do}(\boldsymbol{X}_{I_i}=\boldsymbol{\xi}_{I_i})}\big)_i$. The i'th structural equation of the induced SCM $\mathcal{M}_\mathcal{E}$ is defined as:

$$X_i = h_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{M}_\mathcal{E}}(i)}).$$

Note that this equation is *equivalent* to the labeled equation $\mathcal{E}_i$ in the sense that they have identical solution sets $\{(X_i^*, \boldsymbol{X}^*_{\mathrm{pa}_{\mathcal{M}_\mathcal{E}}(i)})\}$. Repeating this procedure for all $i \in \mathcal{I}$, we obtain the induced SCM $\mathcal{M}_\mathcal{E}$.

This construction is designed to preserve the important mathematical structure. In particular:

Lemma 1 *Let $\mathcal{E}$ be an LEE, $I \subseteq \mathcal{I}$ and $\boldsymbol{\xi}_I \in \mathcal{R}_I$ and consider the perfect intervention $\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)$. Suppose that both the LEE $\mathcal{E}$ and the intervened LEE $\mathcal{E}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ are structurally solvable. (i) Applying the intervention $\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)$ to the induced SCM $\mathcal{M}_\mathcal{E}$ gives the same result as constructing the SCM corresponding to the intervened LEE $\mathcal{E}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$:*

$$(\mathcal{M}_\mathcal{E})_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)} = \mathcal{M}_{\mathcal{E}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}}.$$

(ii) Solvability of the intervened LEE $\mathcal{E}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ implies solvability of the induced intervened SCM $\mathcal{M}_{\mathcal{E}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}}$ and their respective solutions $\boldsymbol{X}^*_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ are identical.

Proof. The first statement directly follows from the construction of the induced SCM. The key observation regarding solvability is the following. From the construction above it directly follows that $\forall i \in \mathcal{I}$:

$$\forall_{\boldsymbol{X}_{\mathrm{pa}_\mathcal{E}(i)} \in \mathcal{R}_{\mathrm{pa}_\mathcal{E}(i)}} : \quad 0 = g_i(\boldsymbol{X}_{\mathrm{pa}_\mathcal{E}(i)}) \iff X_i = h_i(\boldsymbol{X}_{\mathrm{pa}_\mathcal{E}(i)\setminus\{i\}}).$$

This trivially implies:

$$\forall_{\boldsymbol{X}\in\mathcal{R}_\mathcal{I}} \forall_{i\in\mathcal{I}} : 0 = g_i(\boldsymbol{X}_{\mathrm{pa}_\mathcal{E}(i)}) \iff X_i = h_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{M}_\mathcal{E}}(i)}).$$

This means that each simultaneous solution of all labeled equations is a simultaneous solution of all structural equations, and vice versa:

$$\forall_{\boldsymbol{X}\in\mathcal{R}_\mathcal{I}} : \quad \Big(\big[\forall_{i\in\mathcal{I}} : 0 = g_i(\boldsymbol{X}_{\mathrm{pa}_\mathcal{E}(i)})\big] \iff \big[\forall_{i\in\mathcal{I}} : X_i = h_i(\boldsymbol{X}_{\mathrm{pa}_{\mathcal{M}_\mathcal{E}}(i)})\big] \Big).$$

The crucial point is that this still holds if an intervention replaces some of the equations (by $0 = X_i - \xi_i$ and $X_i = \xi_i$, respectively, for all $i \in I$). □

4.5 From ODEs to deterministic SCMs

We can now combine all the results and definitions so far to construct a deterministic SCM from an ODE under certain stability conditions. We define:

Definition 10 *An ODE $\mathcal{D}$ is called* structurally stable *if for each $i \in \mathcal{I}$, the ODE $\mathcal{D}$ is stable with respect to $\{\mathcal{I} \setminus \{i\}\}_{i\in\mathcal{I}}$.*

Consider the diagram in Figure 4. Here, the labels of the arrows correspond with the numbers of the sections that discuss the corresponding mapping. The downward mappings correspond with a particular intervention $\mathrm{do}(\boldsymbol{X}_I = \boldsymbol{\xi}_I)$, applied at the different levels (ODE, induced LEE, induced SCM). Our main result:

Theorem 2 *If both the ODE $\mathcal{D}$ and the intervened ODE $\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ are structurally stable, then: (i) The diagram in Figure 4 commutes.*[5] *(ii) If furthermore, the intervened ODE $\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ is stable, the induced intervened SCM $\mathcal{M}_{\mathcal{E}_{\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}}}$ has a unique solution that coincides with the stable equilibrium of the intervened ODE $\mathcal{D}_{\mathrm{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$.*

Proof. Immediate from Theorem 1 and Lemma 1. □

Note that even though the ODE may contain self-loops (i.e., the time derivative $\dot{X}_i$ could depend on X_i itself, and hence $i \in \mathrm{pa}_\mathcal{D}(i)$), the induced SCM $\mathcal{M}_{\mathcal{E}_\mathcal{D}}$ does *not* contain self-loops by construction (i.e., $i \notin \mathrm{pa}_{\mathcal{M}_{\mathcal{E}_\mathcal{D}}}(i)$). Somewhat surprisingly, the structural stability conditions actually imply the existence of self-loops (because if X_i would not occur in the equilibrium equation $(\mathcal{E}_\mathcal{D})_i$, its value would be undetermined and hence the equilibrium would not be unique).

Whether one prefers the SCM representation over the LEE representation is mainly a matter of practical considerations: both representations contain all the necessary information to predict the results of arbitrary perfect interventions, and one can easily go from the LEE representation to the SCM representation. One can also easily go in the opposite direction, but this cannot be done in a unique way. For example, one could rewrite each structural equation $X_i = h_i(\boldsymbol{X}_{\mathrm{pa}_\mathcal{M}(i)})$ as the equilibrium equation $0 = h_i(\boldsymbol{X}_{\mathrm{pa}_\mathcal{M}(i)}) - X_i$, but also as the equilibrium equation $0 = h_i^3(\boldsymbol{X}_{\mathrm{pa}_\mathcal{M}(i)}) - X_i^3$ (in both cases, it would be given the label i).

In case the dynamics contains no directed cycles (not considering self-loops), the advantage of the SCM representation is that it is more explicit. Starting at the variables without parents, and following the topological ordering of the corresponding directed acyclic

[5]This means that it does not matter in which direction one follows the arrows, the end result will be the same.

$$\begin{tikzcd}
\boxed{\begin{matrix}\text{ODE}\\ \mathcal{D}\end{matrix}} \arrow[r, mapsto, "3.3"] \arrow[d, mapsto, "2.2"] & \boxed{\begin{matrix}\text{LEE}\\ \mathcal{E}_{\mathcal{D}}\end{matrix}} \arrow[r, mapsto, dashed, "4.4"] \arrow[d, mapsto, "3.2"] & \boxed{\begin{matrix}\text{SCM}\\ \mathcal{M}_{\mathcal{E}_{\mathcal{D}}}\end{matrix}} \arrow[d, mapsto, "4.2"] \\
\boxed{\begin{matrix}\text{intervened ODE}\\ \mathcal{D}_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}\end{matrix}} \arrow[r, mapsto, "3.3"] & \boxed{\begin{matrix}\text{intervened LEE}\\ \mathcal{E}_{\mathcal{D}_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}}\end{matrix}} \arrow[r, mapsto, dashed, "4.4"] & \boxed{\begin{matrix}\text{intervened SCM}\\ \mathcal{M}_{\mathcal{E}_{\mathcal{D}_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}}}\end{matrix}}
\end{tikzcd}$$

Figure 4: Each of the arrows in the diagram corresponds with a mapping that is described in the section that the label refers to. The dashed arrows are only defined under structural solvability assumptions on the LEE (or structural stability assumptions on the inducing ODE). If the ODE $\mathcal{D}$ and intervened ODE $\mathcal{D}_{\text{do}(\boldsymbol{X}_I=\boldsymbol{\xi}_I)}$ are structurally stable, this diagram commutes (cf. Theorem 2).

graph, we directly obtain the solution of an SCM by simple substitution in a finite number of steps. When using the LEE representation, one needs to solve a set of equations instead. In the cyclic case, one needs to solve a set of equations in both representations, and the difference is merely cosmetical. However, one could argue that the LEE representation is slightly more natural in the cyclic case, as it does not force us to make additional (structural) stability assumptions.

4.5.1 Example: damped harmonic oscillators

Figure 5 shows the graph of the structural causal model induced by our construction. It reflects the intuition that at equilibrium, (the position of) each mass has a direct causal influence on (the positions of) its neighbors. Observing that the momentum variables always vanish at equilibrium (even for any perfect intervention that we consider), we can decide that the only relevant variables for the SCM are the position variables Q_i. Then, we end up with the following structural equations:

$$Q_i = \frac{k_i(Q_{i+1} - l_i) + k_{i-1}(Q_{i-1} + l_{i-1})}{k_i + k_{i+1}}. \tag{13}$$

5 Discussion

In many empirical sciences (physics, chemistry, biology, etc.) and in engineering, differential equations are a common modeling tool. When estimating system characteristics from data, they are especially useful if measurements can be done on the relevant time scale. If equilibration time scales become too small

$$\begin{tikzcd}
X_1 \arrow[r, bend left] & X_2 \arrow[l, bend left] \arrow[r, bend left] & X_3 \arrow[l, bend left] \arrow[r, bend left] & X_4 \arrow[l, bend left]
\end{tikzcd}$$

Figure 5: Graph of the structural causal model induced by the mass-spring system for $D = 4$.

with respect to the temporal resolution of measurements, however, the more natural representation may be in terms of structural causal models. The main contribution of this work is to build an explicit bridge from the world of differential equations to the world of causal models Our hope is that this may aid in broadening the impact of causal modeling.

Note that information is lost when going from a dynamical system representation to an equilibrium representation (either LEE or SCM), in particular the rate of convergence toward equilibrium. If time-series data is available, the most natural representation may be the dynamical system representation. If only snapshot data or equilibrium data is available, the dynamical system representation can be considered to be overly complicated, and one may use the LEE or SCM representation instead.

We have shown one particular way in which structural causal models can be "derived". We do not claim that this is the only way: on the contrary, SCMs can probably be obtained in several other ways and from other representations as well. A recent example is the derivation of SCMs from stochastic differential equations (Sokol and Hansen, 2013). Other related work on differential equations and causality is (Voortman et al., 2010; Iwasaki and Simon, 1994).

We intend to extend the basic framework described here towards the more general stochastic case. Uncertainty or "noise" can enter in different ways: via uncertainty about (constant) parameters of the differential equations, via uncertainty about the initial condition (in the case of constants of motion) and via latent variables (in the case of confounding).

Acknowledgements

We thank Bram Thijssen, Tom Claassen, Tom Heskes and Tjeerd Dijkstra for stimulating discussions. JM was supported by NWO, the Netherlands Organization for Scientific Research (VENI grant 639.031.036).

References

Bollen, K. A. (1989). *Structural Equations with Latent Variables*. John Wiley & Sons.

Dash, D. (2005). Restructuring dynamic causal systems in equilibrium. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS 2005)*.

Hyttinen, A., Eberhardt, F., and Hoyer, P. (2012). Learning linear cyclic causal models with latent variables. *Journal for Machine Learning Research*, 13:33873439.

Itani, S., Ohannessian, M., Sachs, K., Nolan, G. P., and Dahleh, M. A. (2010). Structure learning in causal cyclic networks. In *JMLR Workshop and Conference Proceedings*, volume 6, page 165176.

Iwasaki, Y. and Simon, H. A. (1994). Causality and model abstraction. *Artificial Intelligence*, 67:143–194.

Koster, J. T. A. (1996). Markov properties of nonrecursive causal models. *Annals of Statistics*, 24(5):2148–2177.

Lacerda, G., Spirtes, P., Ramsey, J., and Hoyer, P. O. (2008). Discovering cyclic causal models by independent components analysis. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-2008)*.

Lauritzen, S. (1996). *Graphical models*. Clarendon Press.

Mooij, J. M., Janzing, D., Heskes, T., and Schölkopf, B. (2011). On causal discovery with cyclic additive noise models. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 24 (NIPS*2011)*, pages 639–647.

Murray, J. (2002). *Mathematical Biology. I: An Introduction*. Springer, 3 edition.

Neal, R. (2000). On deducing conditional independence from d -separation in causal graphs with feedback. *Journal of Artificial Intelligence Research*, 12:87–91.

Pearl, J. (2000). *Causality*. Cambridge University Press.

Pearl, J. and Dechter, R. (1996). Identifying independence in causal graphs with feedback. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI-96)*, pages 420–426.

Richardson, T. (1996). A discovery algorithm for directed cyclic graphs. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence (UAI-1996)*.

Schmidt, M. and Murphy, K. (2009). Modeling discrete interventional data using directed cyclic graphical models. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI-09)*.

Sokol, A. and Hansen, N. R. (2013). Causal interpretation of stochastic differential equations. *arXiv.org preprint*, arXiv:1304.0217 [math.PR].

Spirtes, P. (1995). Directed cyclic graphical representations of feedback models. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence (UAI-95)*, page 491499.

Spirtes, P., Glymour, C., and Scheines, R. (1993). *Causation, prediction, and search*. Springer-Verlag. (2nd edition MIT Press 2000).

Voortman, M., Dash, D., and Druzdzel, M. (2010). Learning why things change: The difference-based causality learner. In *Proceedings of the Twenty-Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 641–650, Corvallis, Oregon. AUAI Press.

One-Class Support Measure Machines for Group Anomaly Detection

Krikamol Muandet
Empirical Inference Department
Max Planck Institute for Intelligent Systems
Spemannstraße 38, 72076 Tübingen

Bernhard Schölkopf
Empirical Inference Department
Max Planck Institute for Intelligent Systems
Spemannstraße 38, 72076 Tübingen

Abstract

We propose one-class support measure machines (OCSMMs) for group anomaly detection. Unlike traditional anomaly detection, OCSMMs aim at recognizing anomalous aggregate behaviors of data points. The OCSMMs generalize well-known one-class support vector machines (OCSVMs) to a space of probability measures. By formulating the problem as quantile estimation on distributions, we can establish interesting connections to the OCSVMs and variable kernel density estimators (VKDEs) over the input space on which the distributions are defined, bridging the gap between large-margin methods and kernel density estimators. In particular, we show that various types of VKDEs can be considered as solutions to a class of regularization problems studied in this paper. Experiments on Sloan Digital Sky Survey dataset and High Energy Particle Physics dataset demonstrate the benefits of the proposed framework in real-world applications.

1 Introduction

Anomaly detection is one of the most important tools in all data-driven scientific disciplines. Data that do not conform to the expected behaviors often bear some interesting characteristics and can help domain experts better understand the problem at hand. However, in the era of data explosion, the anomaly may appear not only in the data themselves, but also as a result of their interactions. The main objective of this paper is to investigate the latter type of anomalies. To be consistent with the previous works (Póczos et al. 2011, Xiong et al. 2011a;b), we will refer to this problem as a group anomaly detection, as opposed to a traditional point anomaly detection.

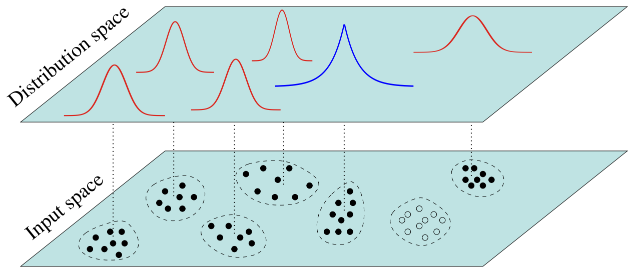


Figure 1: An illustration of two types of group anomalies. An anomalous group may be a group of anomalous samples which is easy to detect (unfilled points). In this paper, we are interested in detecting anomalous groups of normal samples (filled points) which is more difficult to detect because of the higher-order statistics. Note that group anomaly we are interested in can only be observed in the space of distributions.

Like traditional point anomaly detection, the group anomaly detection refers to a problem of finding patterns in groups of data that do not conform to expected behaviors (Póczos et al. 2011, Xiong et al. 2011a;b). That is, an ultimate goal is to detect interesting aggregate behaviors of data points among several groups. In principle, anomalous groups may consist of individually anomalous points, which are relatively easy to detect. On the other hand, anomalous groups of relatively normal points, whose behavior as a group is unusual, is much more difficult to detect. In this work, we are interested in the latter type of group anomalies. Figure 1 illustrates this scenario.

Group anomaly detection may shed light in a wide range of applications. For example, a Sloan Digital Sky Survey (SDSS) has produced a tremendous amount of astronomical data. It is therefore very crucial to detect rare objects such as stars, galaxies, or quasars that might lead to a scientific discovery. In addition to individual celestial objects, investigating groups of them may help astronomers understand the universe on larger scales. For instance, the anomalous group of galaxies, which is the smallest aggregates of galaxies, may reveal interesting phenomena, e.g., the gravitational interactions of galaxies.

Likewise, a new physical phenomena in high energy particle physics such as Higgs boson appear as a tiny excesses of certain types of collision events among a vast background of known physics in particle detectors (Bhat 2011, Vatanen et al. 2012). Investigating each collision event individually is no longer sufficient as the individual events may not be anomalies by themselves, but their occurrence together as a group is anomalous. Hence, we need a powerful algorithm to detect such a rare and highly structured anomaly.

Lastly, the algorithm proposed in this paper can be applied to point anomaly detection with substantial and heterogeneous uncertainties. For example, it is often costly and time-consuming to obtain the full spectra of astronomical objects. Instead, relatively noisier measurements are usually made. In addition, the estimated uncertainty which represents the uncertainty one would obtain from multiple observations is also available. Incorporating these uncertainties has been shown to improve the performance of the learning systems (Bovy et al. 2011, Kirkpatrick et al. 2011, Ross et al. 2012).

The anomaly detection has been intensively studied (Chandola et al. (2009) and references therein). However, few attempts have been made on developing successful group anomaly detection algorithms. For example, a straightforward approach is to define a set of features for each group and apply standard point anomaly detection (Chan and Mahoney 2005). Despite its simplicity, this approach requires a specific domain knowledge to construct appropriate sets of features. Another possibility is to first identify the individually anomalous points and then find their aggregations (Das et al. 2008). Again, this approach relies only on the detection of anomalous points and thus cannot find the anomalous groups in which their members are perfectly normal. Successful group anomaly detectors should be able to incorporate the higher-order statistics of the groups.

Recently, a family of hierarchical probabilistic models based on a Latent Dirichlet Allocation (LDA) (Blei et al. 2003) has been proposed to cope with both types of group anomalies (Xiong et al. 2011a;b). In these models, the data points in each group are assumed to be one of the K different types and generated by a mixture of K Gaussian distributions. Although the distributions over these K types can vary across M groups, they share common generator. The groups that have small probabilities under the model are marked as anomalies using scoring criteria defined as a combination of a point-based anomaly score and a group-based anomaly score. The Flexible Genre Model (FGM) recently extends this idea to model more complex group structures (Xiong et al. 2011a).

Instead of employing a generative approach, we propose a simple and efficient discriminative way of detecting group anomaly. In this work, M groups of data points are represented by a set of M probability distributions assumed to be i.i.d. realization of some unknown distribution $\mathscr{P}$. In practice, only i.i.d samples from these distributions are observed. Hence, we can treat group anomaly detection as detecting the anomalous distributions based on their empirical samples. To allow for a practical algorithm, the distributions are mapped into the reproducing kernel Hilbert space (RKHS) using the kernel mean embedding. By working directly with the distributions, the higher-order information arising from the aggregate behaviors of the data points can be incorporated efficiently.

2 Quantile Estimation on Probability Distributions

Let $\mathcal{X}$ denote a non-empty input space with associated σ-algebra $\mathcal{A}$, $\mathbb{P}$ denote the probability distribution on $(\mathcal{X}, \mathcal{A})$, and $\mathfrak{P}_{\mathcal{X}}$ denote the set of all probability distributions on $(\mathcal{X}, \mathcal{A})$. The space $\mathfrak{P}_{\mathcal{X}}$ is endowed with the topology of weak convergence and the associated Borel σ-algebra.

We assume that there exists a distribution $\mathscr{P}$ on $\mathfrak{P}_{\mathcal{X}}$, where $\mathbb{P}_1, \ldots, \mathbb{P}_\ell$ are i.i.d. realizations from $\mathscr{P}$, and the sample S_i is made of n_i i.i.d. samples distributed according to the distribution $\mathbb{P}_i$. In this work, we observe ℓ samples $S_i = \{x_k^{(i)}\}_{1 \le k \le n_i}$ for $i = 1, \ldots, \ell$. For each sample S_i, $\widehat{\mathbb{P}}_i = \frac{1}{n_i}\sum_{j=1}^{n_i} \delta_{x_j^{(i)}}$ is the associated empirical distribution of $\mathbb{P}_i$.

In this work, we formulate a group anomaly detection problem as learning quantile function $q : \mathfrak{P}_{\mathcal{X}} \to \mathbb{R}$ to estimate the support of $\mathscr{P}$. Let $\mathcal{C}$ be a class of measurable subsets of $\mathfrak{P}_{\mathcal{X}}$ and λ be a real-valued function defined on $\mathcal{C}$, the quantile function w.r.t. $(\mathscr{P}, \mathcal{C}, \lambda)$ is

$$q(\beta) = \inf\{\lambda(C) : \mathscr{P}(C) \ge \beta, C \in \mathcal{C}\} \ ,$$

where $0 < \beta \le 1$. In this paper, we consider when λ is Lebesgue measure, in which case $C(\beta)$ is the minimum volume $C \in \mathcal{C}$ that contains at least a fraction β of the probability mass of $\mathscr{P}$. Thus, the function q can be used to test if any test distribution $\mathbb{P}_t$ is anomalous w.r.t. the training distributions.

Rather than estimating $C(\beta)$ in the space of distributions directly, we first map the distributions into a feature space via a positive semi-definite kernel k.

Our class $\mathcal{C}$ is then implicitly defined as the set of half-spaces in the feature space. Specifically, $C_{\mathbf{w}} = \{\mathbb{P} \mid f_{\mathbf{w}}(\mathbb{P}) \geq \rho\}$ where $(\mathbf{w}, \rho)$ are respectively a weight vector and an offset parametrizing a hyperplane in the feature space associated with the kernel k. The optimal $(\mathbf{w}, \rho)$ is obtained by minimizing a regularizer which controls the smoothness of the estimated function describing C.

3 One-Class Support Measure Machines

In order to work with the probability distributions efficiently, we represent the distributions as mean functions in a reproducing kernel Hilbert space (RKHS) (Berlinet and Agnan 2004, Smola et al. 2007). Formally, let $\mathcal{H}$ denote an RKHS of functions $f : \mathcal{X} \to \mathbb{R}$ with reproducing kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The kernel mean map from $\mathfrak{P}_{\mathcal{X}}$ into $\mathcal{H}$ is defined as

$$\mu : \mathfrak{P}_{\mathcal{X}} \to \mathcal{H}, \;\; \mathbb{P} \longmapsto \int_{\mathcal{X}} k(x, \cdot) \, \mathrm{d}\mathbb{P}(x) \;\; . \qquad (1)$$

We assume that $k(x, \cdot)$ is bounded for any $x \in \mathcal{X}$. For any $\mathbb{P}$, letting $\mu_{\mathbb{P}} = \mu(\mathbb{P})$, one can show that $\mathbb{E}_{\mathbb{P}}[f] = \langle \mu_{\mathbb{P}}, f \rangle_{\mathcal{H}}$, for all $f \in \mathcal{H}$.

The following theorem due to Fukumizu et al. (2004) and Sriperumbudur et al. (2010) gives a promising property of representing distributions as mean elements in the RKHS.

Theorem 1. *The kernel k is characteristic if and only if the map* (1) *is injective.*

Examples of characteristic kernels include Gaussian RBF kernel and Laplace kernel. Using the characteristic kernel k, Theorem 1 implies that the map (1) preserves all information about the distributions. Hence, one can apply many existing kernel-based learning algorithms to the distributions as if they are individual samples with no information loss.

Intuitively, one may view the mean embeddings of the distributions as their feature representations. Thus, our approach is in line with previous attempts in group anomaly detection that find a set of appropriate features for each group. On the one hand, however, the mean embedding approach captures all necessary information about the groups without relying heavily on a specific domain knowledge. On the other hand, it is flexible to choose the feature representation that is suitable to the problem at hand via the choice of the kernel k.

3.1 OCSMM Formulation

Using the mean embedding representation (1), the primal optimization problem for one-class SMM can be subsequently formulated in an analogous way to the one-class SVM (Schölkopf et al. 2001) as follow:

$$\underset{\mathbf{w},b,\xi,\rho}{\text{minimize}} \quad \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle_{\mathcal{H}} - \rho + \frac{1}{\nu\ell}\sum_{i=1}^{\ell} \xi_i \qquad (2a)$$

$$\text{subject to} \quad \langle \mathbf{w}, \mu_{\mathbb{P}_i} \rangle_{\mathcal{H}} \geq \rho - \xi_i, \xi_i \geq 0 \qquad (2b)$$

where ξ_i denote slack variables and $\nu \in (0, 1]$ is a trade-off parameter corresponding to an expected fraction of outliers within the feature space. The trade-off ν is an upper bound on the fraction of outliers and lower bound on the fraction of support measures (Schölkopf et al. 2001).

The trade-off parameter ν plays an important role in group anomaly detection. Small ν implies that anomalous groups are rare compared to the normal groups. Too small ν leads to some anomalous groups being rejected. On the other hand, large ν implies that anomalous groups are common. Too large ν leads to some normal groups being accepted as anomaly. As group anomaly is subtle, one need to choose ν very carefully to reduce the effort in the interpretation of the results.

By introducing Lagrange multipliers $\boldsymbol{\alpha}$, we have $\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i \mu_{\mathbb{P}_i} = \sum_{i=1}^{\ell} \alpha_i \mathbb{E}_{\mathbb{P}_i}[k(x, \cdot)]$ and the dual form of (2) can be written as

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \alpha_i \alpha_j \langle \mu_{\mathbb{P}_i}, \mu_{\mathbb{P}_j} \rangle_{\mathcal{H}} \qquad (3a)$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq \frac{1}{\nu\ell}, \; \sum_{i=1}^{\ell} \alpha_i = 1 \, . \qquad (3b)$$

Note that the dual form is a quadratic programming and depends on the inner product $\langle \mu_{\mathbb{P}_i}, \mu_{\mathbb{P}_j} \rangle_{\mathcal{H}}$. Given that we can compute $\langle \mu_{\mathbb{P}_i}, \mu_{\mathbb{P}_j} \rangle_{\mathcal{H}}$, we can employ the standard QP solvers to solve (3).

3.2 Kernels on Probability Distributions

From (3), we can see that $\mu_{\mathbb{P}}$ is a feature map associated with the kernel $K : \mathfrak{P}_{\mathcal{X}} \times \mathfrak{P}_{\mathcal{X}} \to \mathbb{R}$, defined as $K(\mathbb{P}_i, \mathbb{P}_j) = \langle \mu_{\mathbb{P}_i}, \mu_{\mathbb{P}_j} \rangle_{\mathcal{H}}$. It follows from Fubini's theorem and reproducing property of $\mathcal{H}$ that

$$\begin{aligned} \langle \mu_{\mathbb{P}_i}, \mu_{\mathbb{P}_j} \rangle_{\mathcal{H}} &= \iint \langle k(x, \cdot), k(y, \cdot) \rangle_{\mathcal{H}} \, \mathrm{d}\mathbb{P}_i(x) \, \mathrm{d}\mathbb{P}_j(y) \\ &= \iint k(x, y) \, \mathrm{d}\mathbb{P}_i(x) \, \mathrm{d}\mathbb{P}_j(y) \;\; . \qquad (4) \end{aligned}$$

Hence, K is a positive definite kernel on $\mathfrak{P}_{\mathcal{X}}$. Given the sample sets $S_1, \ldots, S_\ell$, one can estimate (4) by

$$K(\widehat{\mathbb{P}}_i, \widehat{\mathbb{P}}_j) = \frac{1}{n_i \cdot n_j} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} k(x_k^{(i)}, x_l^{(j)}) \qquad (5)$$

where $x_k^{(i)} \in S_i$, $x_l^{(j)} \in S_j$, and n_i is the number of samples in S_i for $i = 1, \ldots, \ell$.

Previous works in kernel-based anomaly detection have shown that the Gaussian RBF kernel is more suitable than some other kernels such as polynomial kernels (Hoffmann 2007). Thus we will focus primarily on the Gaussian RBF kernel given by

$$k_\sigma(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right), \;\; x, x' \in \mathcal{X} \qquad (6)$$

where $\sigma > 0$ is a bandwidth parameter. In the sequel, we denote the reproducing kernel Hilbert space associated with kernel k_σ by $\mathcal{H}_\sigma$. Also, let $\Phi : \mathcal{X} \to \mathcal{H}_\sigma$ be a feature map such that $k_\sigma(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}_\sigma}$.

In group anomaly detection, we always observe the i.i.d. samples from the distribution underlying the group. Thus, it is natural to use the empirical kernel (5). However, one may relax this assumption and apply the kernel (4) directly. For instance, if we have a Gaussian distribution $\mathbb{P}_i = \mathcal{N}(m_i, \Sigma_i)$ and a Gaussian RBF kernel k_σ, we can compute the kernel analytically by

$$K(\mathbb{P}_i, \mathbb{P}_j) = \frac{\exp\left(-\frac{1}{2}(m_i - m_j)^\top B^{-1}(m_i - m_j)\right)}{|\frac{1}{\sigma^2}\Sigma_i + \frac{1}{\sigma^2}\Sigma_j + \mathbf{I}|^{\frac{1}{2}}} \qquad (7)$$

where $B = \Sigma_i + \Sigma_j + \sigma^2\mathbf{I}$. This kernel is particularly useful when one want to incorporate the point-wise uncertainty of the observation into the learning algorithm (Muandet et al. 2012). More details will be given in Section 4.2 and 5).

4 Theoretical Analysis

This section presents some theoretical analyses. The geometrical interpretation of OCSMMs is given in Section 4.1. Then, we discuss the connection of OCSMM to the kernel density estimator in Section 4.2. In the sequel, we will focus on the translation-invariant kernel function to simplify the analysis.

4.1 Geometric Interpretation

For translation-invariant kernel, $k(x, x)$ is constant for all $x \in \mathcal{X}$. That is, $\|\Phi(x)\|_{\mathcal{H}} = \tau$ for some constant ρ. This implies that all of the images $\Phi(x)$ lie on the sphere in the feature space (cf. Figure 2a). Consequently, the following inequality holds

$$\|\mu_{\mathbb{P}}\|_{\mathcal{H}} = \left\| \int k(x, \cdot)\, \mathrm{d}\mathbb{P}(x) \right\|_{\mathcal{H}} \leq \int \|k(x, \cdot)\|_{\mathcal{H}}\, \mathrm{d}\mathbb{P}(x) = \tau\,,$$

which shows that all mean embeddings lie inside the sphere (cf. Figure 2a). As a result, we can establish the existence and uniqueness of the separating hyperplane $\mathbf{w}$ in (2) through the following theorem.

Theorem 2. *There exists a unique separating hyperplane* $\boldsymbol{w}$ *as a solution to* (2) *that separates* $\mu_{\mathbb{P}_1}, \mu_{\mathbb{P}_2}, \ldots, \mu_{\mathbb{P}_\ell}$ *from the origin.*

Proof. Due to the separability of the feature maps $\Phi(x)$, the convex hull of the mean embeddings $\mu_{\mathbb{P}_1}, \mu_{\mathbb{P}_2}, \ldots, \mu_{\mathbb{P}_\ell}$ does not contain the origin. The existence and uniqueness of the hyperplane then follows from the supporting hyperplane theorem (Schölkopf and Smola 2001). ■

By Theorem 2, the OCSMM is a simple generalization of OCSVM to the space of probability distributions. Furthermore, the straightforward generalization will allow for a direct application of an efficient learning algorithm as well as existing theoretical results.

There is a well-known connection between the solution of OCSVM with translation invariant kernels and the center of the minimum enclosing sphere (MES) (Tax and Duin 1999; 2004). Intuitively, this is not the case for OCSMM, even when the kernel k is translation-invariant, as illustrated in Figure 2b. Fortunately, the connection between OCSMM and MES can be made precise by applying the spherical normalization

$$\langle \mu_{\mathbb{P}}, \mu_{\mathbb{Q}} \rangle_{\mathcal{H}} \longmapsto \frac{\langle \mu_{\mathbb{P}}, \mu_{\mathbb{Q}} \rangle_{\mathcal{H}}}{\sqrt{\langle \mu_{\mathbb{P}}, \mu_{\mathbb{P}} \rangle_{\mathcal{H}} \langle \mu_{\mathbb{Q}}, \mu_{\mathbb{Q}} \rangle_{\mathcal{H}}}} \qquad (8)$$

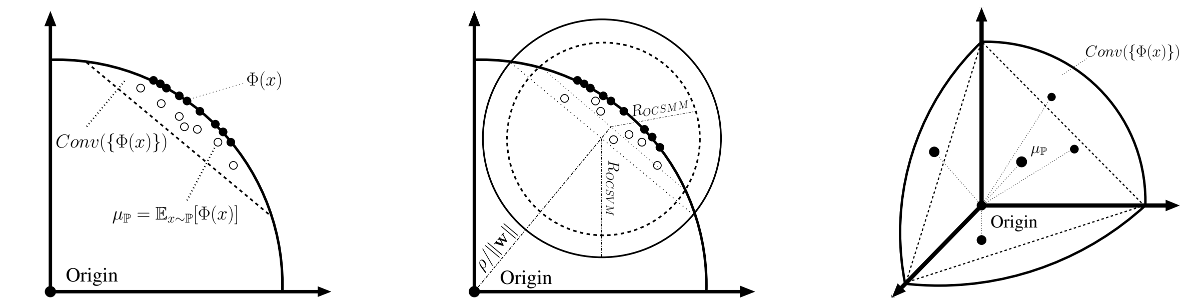
After the normalization, $\|\mu_{\mathbb{P}}\|_{\mathcal{H}} = 1$ for all $\mathbb{P} \in \mathfrak{P}_{\mathcal{X}}$. That is, all mean embeddings lie on the unit sphere in the feature space. Consequently, the OCSMM and MES are equivalent after the normalization.

Given the equivalence between OCSMM and MES, it is natural to ask if the spherical normalization (8) preserves the injectivity of the Hilbert space embedding. In other words, is there an information loss after the normalization? The following theorem answers this question for kernel k that satisfies some reasonable assumptions.

Theorem 3. *Assume that k is characteristic and the samples are linearly independent in the feature space $\mathcal{H}$. Then, the spherical normalization preserves the injectivity of the mapping $\mu : \mathfrak{P}_{\mathcal{X}} \to \mathcal{H}$.*

Proof. Let us assume the normalization does not preserve the injectivity of the mapping. Thus, there exist two distinct probability distributions $\mathbb{P}$ and $\mathbb{Q}$ for which

$$\begin{aligned} \mu_{\mathbb{P}} &= \mu_{\mathbb{Q}} \\ \int k(x, \cdot)\, \mathrm{d}\mathbb{P}(x) &= \int k(x, \cdot)\, \mathrm{d}\mathbb{Q}(x) \\ \int k(x, \cdot)\, \mathrm{d}(\mathbb{P} - \mathbb{Q})(x) &= 0\;. \end{aligned}$$



(a) feature map and mean map (b) minimum enclosing sphere (c) spherical normalization

Figure 2: (a) The two dimensional representation of the RKHS of Gaussian RBF kernels. Since the kernels depend only on $x - x'$, $k(x, x)$ is constant. Therefore, all feature maps $\Phi(x)$ (black dots) lie on a sphere in feature space. Hence, for any probability distribution $\mathbb{P}$, its mean embedding $\mu_{\mathbb{P}}$ always lies in the convex hull of the feature maps, which in this case, forms a segment of the sphere. (b) In general, the solution of OCSMM is different from the minimum enclosing sphere. (c) Three dimensional sphere in the feature space. For the Gaussian RBF kernel, the kernel mean embeddings of all distributions always lie inside the segment of the sphere. In addition, the angle between any pair of mean embeddings is always greater than zero. Consequently, the mean embeddings can be scaled, e.g., to lie on the sphere, and the map is still injective.

As $\mathbb{P} \neq \mathbb{Q}$, the last equality holds if and only if there exists $x \in \mathcal{X}$ for which $k(x, \cdot)$ are linearly dependent, which contradicts the assumption. Consequently, the spherical normalization must preserve the injectivity of the mapping. ■

The Gaussian RBF kernel satisfies the assumption given in Theorem 3 as the kernel matrix will be full-rank and thereby the samples are linearly independent in the feature space. Figure 2c depicts an effect of the spherical normalization.

It is important to note that the spherical normalization does not necessarily improve the performance of the OCSMM. It ensures that all the information about the distributions are preserved.

4.2 OCSMM and Density Estimation

In this section we make a connection between the OC-SMM and kernel density estimation (KDE). First, we give a definition of the KDE. Let $x_1, x_2, \ldots, x_n$ be an i.i.d. samples from some distribution F with unknown density f, the KDE of f is defined as

$$\hat{f}(y) = \frac{1}{nh}\sum_{i=1}^{n} k\left(\frac{y - x_i}{h}\right) \qquad (9)$$

For $\hat{f}$ to be a density, we require that the kernel satisfies $k(\cdot,\cdot) \geq 0$ and $\int k(x,\cdot)\,\mathrm{d}x = 1$, which includes, for example, the Gaussian kernel, the multivariate Student kernel, and the Laplacian kernel.

When $\nu = 1$, it is well-known that, under some technical assumptions, the OCSVM corresponds exactly to the KDE (Schölkopf et al. 2001). That is, the solution $\mathbf{w}$ of (2) can be written as a uniform sum over training samples similar to (9). Moreover, setting $\nu < 1$ yields a sparse representation where the summand consists of only support vectors of the OCSVM.

Interestingly, we can make a similar correspondence between the KDE and the OCSMM. It follows from Lemma 4 of Muandet et al. (2012) that for certain classes of training probability distributions, the OCSMM on these distributions corresponds to the OCSVM on some training samples equipped with an appropriate kernel function. To understand this connection, consider the OCSMM with the Gaussian RBF kernel k_σ and isotropic Gaussian distributions $\mathcal{N}(m_1;\sigma_1^2), \mathcal{N}(m_2;\sigma_2^2), \ldots, \mathcal{N}(m_n;\sigma_n^2)$[1]. We analyze this scenario under two conditions:

(C1) Identical bandwidth. If $\sigma_i = \sigma_j$ for all $1 \leq i, j \leq n$, the OCSMM is equivalent to the OCSVM on the training samples $m_1, m_2, \ldots, m_n$ with Gaussian RBF kernel $k_{\sigma^2+\sigma_i^2}$ (cf. the kernel (7)). Hence, the OCSMM corresponds to the OCSVM on the means of the distributions with kernel of larger bandwidth.

(C2) Variable bandwidth. Similarly, if $\sigma_i \neq \sigma_j$ for some $1 \leq i, j \leq n$, the OCSMM is equivalent to the OCSVM on the training samples $m_1, m_2, \ldots, m_n$ with Gaussian RBF kernel $k_{\sigma^2+\sigma_i^2}$. Note that the kernel bandwidth may be different at each training samples. Thus, OCSMM in this case corresponds to the OCSVM with variable bandwidth parameters.

[1]We adopt the Gaussian distributions here for the sake of simplicity. More general statement for non-Gaussian distributions follows straightforwardly.

On the one hand, the above scenario allows the OCSVM to cope with noisy/uncertain inputs, leading to more robust point anomaly detection algorithm. That is, we can treat the means as the measurements and the covariances as the measurement uncertainties (cf. Section 5.2). On the other hand, one can also interpret the OCSMM when $\nu = 1$ as a generalization of traditional KDE, where we have a data-dependent bandwidth at each data point. This type of KDE is known in the statistics as variable kernel density estimators (VKDEs) (Abramson 1982, Breiman et al. 1977, Terrell and Scott 1992). For $\nu < 1$, the OCSMM gives a sparse representation of the VKDE.

Formally, the VKDE is characterized by (9) with an adaptive bandwidth $h(x_i)$. For example, the bandwidth is adapted to be larger where the data are less dense, with the aim to reduce the bias. There are basically two different views of VKDE. The first is known as a *balloon estimator* (Terrell and Scott 1992). Essentially, its bandwidth may depend only on the point at which the estimate is taken, i.e., the bandwidth in (9) may be written as $h(y)$. The second type of VKDE is a *sample smoothing estimator* (Terrell and Scott 1992). As opposed to the balloon estimator, it is a mixture of individually scaled kernels centered at each observation, i.e., the bandwidth is $h(x_i)$. The advantage of balloon estimator is that it has a straightforward asymptotic analysis, but the final estimator may not be a density. The sample smoothing estimator is a density if k is a density, but exhibits *non-locality.*

Both types of the VKDEs may be seen from the OCSMM point of view. Firstly, under the condition **(C1)**, the balloon estimator can be recovered by considering different test distribution $\mathbb{P}_t = \mathcal{N}(m_t; \sigma_t)$. As $\sigma_t \to 0$, one obtain the standard KDE on m_t. Similarly, the OCSMM under the condition **(C2)** with $\mathbb{P}_t = \delta_{m_t}$ gives the sample smoothing estimator. Interestingly, the OCSMM under the condition **(C2)** with $\mathbb{P}_t = \mathcal{N}(m_t; \sigma_t)$ results in a combination of these two types of the VKDEs.

In summary, we show that many variants of KDE can be seen as solutions to the regularization functional (2), and thereby provides an insight into a connection between large-margin approach and kernel density estimation.

5 Experiments

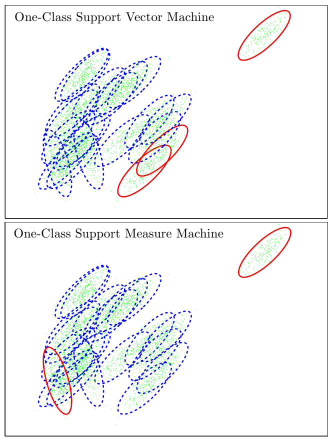
We firstly illustrate a fundamental difference between point and group anomaly detection problems. Then, we demonstrate an advantage of OCSMM on uncertain data when the noise is observed explicitly. Lastly, we compare the OCSMM with existing group anomaly detection techniques, namely, K-nearest neighbor (KNN) based anomaly detection (Zhao and Saligrama 2009) with NP-L_2 divergence and NP-Renyi divergence (Póczos et al. 2011), and Multinomial Genre Model (MGM) (Xiong et al. 2011b) on Sloan Digital Sky Survey (SDSS) dataset and High Energy Particle Physics dataset.

Model Selection and Setup. One of the longstanding problems of one-class algorithms is model selection. Since no labeled data is available during training, we cannot perform cross validation. To encourage a fair comparison of different algorithms in our experiments, we will try out different parameter settings and report the best performance of each algorithm. We believe this simple approach should serve its purpose at reflecting the relative performance of different algorithms. We will employ the Gaussian RBF kernel (6) throughout the experiments. For the OCSVM and the OCSMM, the bandwidth parameter σ^2 is fixed at median$\{\|x_k^{(i)} - x_l^{(j)}\|^2\}$ for all i, j, k, l where $x_k^{(i)}$ denotes the k-th data point in the i-th group, and we consider $\nu = (0.1, 0.2, \ldots, 0.9)$. The OCSVM treats group means as training samples. For synthetic experiments with OCSMM, we use the empirical kernel (5), whereas the non-linear kernel $K(\mathbb{P}_i, \mathbb{P}_j) = \exp(\|\mu_{\mathbb{P}_i} - \mu_{\mathbb{P}_j}\|_{\mathcal{H}}^2/2\gamma^2)$ will be used for real data where we set $\gamma = \sigma$. Our experiments suggest that these choices of parameters usually work well in practice. For KNN-L_2 and KNN-Renyi (α=0.99), we consider when there are 3,5,7,9, and 11 nearest neighbors. For MGM, we follow the same experimental setup as in Xiong et al. (2011b).

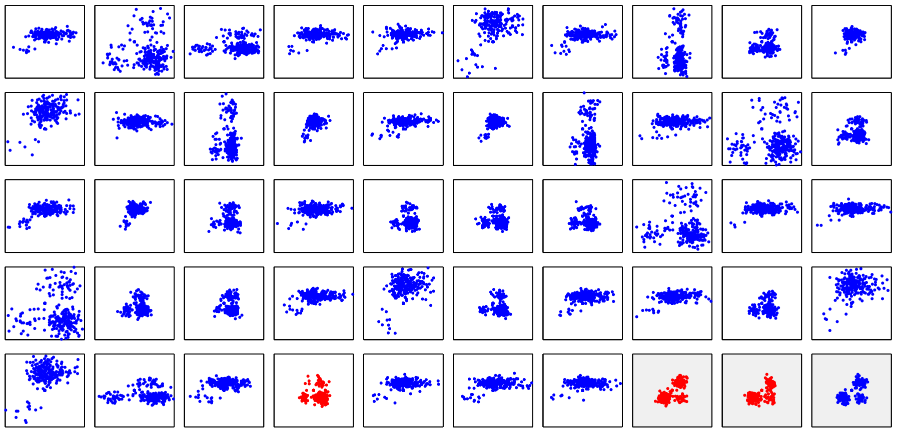
5.1 Synthetic Data

To illustrate the difference between point anomaly and group anomaly, we represent the group of data points by the 2-dimensional Gaussian distribution. We generate 20 normal groups with the covariance $\mathbf{\Sigma} = [0.01, 0.008; 0.008, 0.01]$. The means of these groups are drawn uniformly from $[0, 1]$. Then, we generate 2 anomalous groups of Gaussian distributions whose covariances are rotated by 60 degree from the covariance $\mathbf{\Sigma}$. Furthermore, we perturb one of the normal groups to make it relatively far from the rest of the dataset to introduce an additional degree of anomaly (cf. Figure 3a). Lastly, we generate 100 samples from each of these distributions to form the training set.

For the OCSVM, we represent each group by its empirical average. Since the expected proportion of outliers in the dataset is approximately 10%, we use $\nu = 0.1$ accordingly for both OCSVM and OCSMM. Figure 3a depicts the result which demonstrates that the OCSMM can detect anomalous aggregate patterns undetected by the OCSVM.



(a) OCSVM vs OCSMM



(b) The results of the OCSMM on the mixture of Gaussian dataset

Figure 3: (a) The results of group anomaly detection on synthetic data obtained from the OCSVM and the OCSMM. Blue dashed ovals represent the normal groups, whereas red ovals represent the detected anomalous groups. The OCSVM is only able to detect the anomalous groups that are spatially far from the rest in the dataset, whereas the OCSMM also takes into account other higher-order statistics and therefore can also detect anomalous groups which possess distinctive properties. (b) The results of the OCSMM on the synthetic data of the mixture of Gaussian. The shaded boxes represent the anomalous groups that have different mixing proportion to the rest of the dataset. The OCSMM is able to detects the anomalous groups although they look reasonably normal and cannot be easily distinguished from other groups in the data set based only on an inspection.

Then, we conduct similar experiment as that in Xiong et al. (2011b). That is, the groups are represented as a mixture of four 2-dimensional Gaussian distributions. The means of the mixture components are $[-1,-1],[1,-1],[0,1],[1,1]$ and the covariances are all $\mathbf{\Sigma} = 0.15 \times \mathbf{I}_2$, where $\mathbf{I}_2$ denotes the 2D identity matrix. Then, we design two types of normal groups, which are specified by two mixing proportions $[0.22, 0.64, 0.03, 0.11]$ and $[0.22, 0.03, 0.64, 0.11]$, respectively. To generate a normal group, we first decide with probability $[0.48, 0.52]$ which mixing proportion will be used. Then, the data points are generated from mixture of Gaussian using the specified mixing proportion. The mixing proportion of the anomalous group is $[0.61, 0.1, 0.06, 0.23]$.

We generated 47 normal groups with $n_i \sim \text{Poisson}(300)$ instances in each group. Note that the individual samples in each group are perfectly normal compared to other samples. To test the performance of our technique, we inject the group anomalies, where the individual points are normal, but they together as a group look anomalous. In this anomalous group the individual points are samples from one of the $K = 4$ normal topics, but the mixing proportion was different from both of the normal mixing proportions. We inject 3 anomalous groups into the data set. The OCSMM is trained using the same setting as in the previous experiment. The results are depicted in Figure 3b.

5.2 Noisy Data

As discussed at the end of Section 3.2, the OCSMM may be adopted to learn from data points whose uncertainties are observed explicitly. To illustrate this claim, we generate samples from the unit circle using $x = \cos\theta + \varepsilon$ and $y = \sin\theta + \varepsilon$ where $\theta \sim (-\pi, \pi]$ and ε is a zero-mean isotropic Gaussian noise $\mathcal{N}(0, 0.05)$. A different point-wise Gaussian noise $\mathcal{N}(0, \omega_i)$ where $\omega_i \in (0.2, 0.3)$ is further added to each point to simulate the random measurement corruption. In this experiment, we assume that ω_i is available during training. This situation is often encountered in many applications such as astronomy and computational biology. Both OCSVM and OCSMM are trained on the corrupted data. As opposed to the OCSVM that considers only the observed data points, the OCSMM also uses ω_i for every point via the kernel (7). Then, we consider a slightly more complicate data generated by $x = r \cdot \cos(\theta)$ and $y = r \cdot \sin(\theta)$ where $r = \sin(4\theta) + 2$ and $\theta \in (0, 2\pi]$. The data used in both examples are illustrated in Figure 4.

As illustrated by Figure 4, the density function estimated by the OCSMM is relatively less susceptible to the additional corruption than that estimated by the OCSVM, and tends to estimate the true density more accurately. This is not surprising because we also take into account an additional information about the uncertainty. However, this experiment suggests that when dealing with uncertain data, it might be ben-

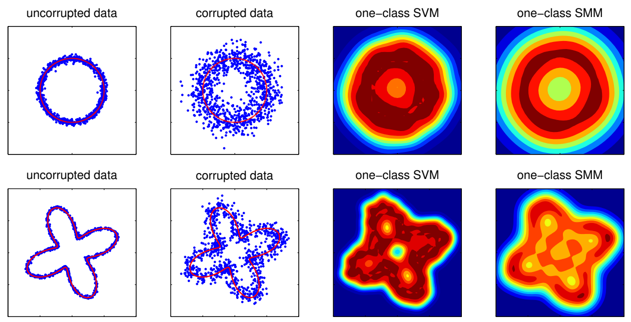


Figure 4: The density functions estimated by the OCSVM and the OCSMM using the corrupted data.

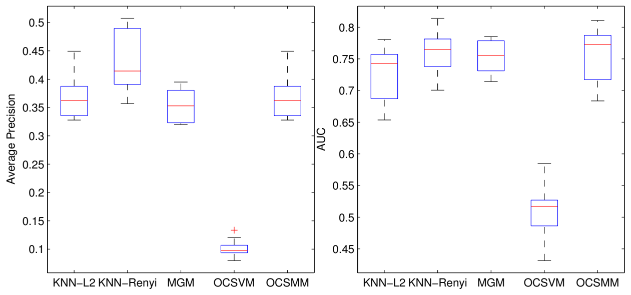


Figure 5: The average precision (AP) and area under the ROC curve (AUC) of different group anomaly detection algorithms on the SDSS dataset.

eficial to also estimate the uncertainty, as commonly performed in astronomy, and incorporate it into the model. This scenario has not been fully investigated in AI and machine learning communities. Our framework provides one possible way to deal with such a scenario.

5.3 Sloan Digital Sky Survey

Sloan Digital Sky Survey (SDSS)[2] consists of a series of massive spectroscopic surveys of the distant universe, the milky way galaxies, and extrasolar planetary systems. The SDSS datasets contain images and spectra of more than 930,000 galaxies and more than 120,000 quasars.

In this experiment, we are interested in identifying anomalous groups of galaxies, as previously studied in Póczos et al. (2011) and Xiong et al. (2011a;b). To replicate the experiments conducted in Xiong et al. (2011b), we use the same dataset which consists of 505 spatial clusters of galaxies. Each of which contains about 10-15 galaxies. The data were preprocessed by PCA to reduce the 1000-dimensional features to 4-dimensional vectors.

To evaluate the performance of different algorithms to detect group anomaly, we consider artificially random injections. Each anomalous group is constructed by randomly selecting galaxies. There are 50 anomalous groups of galaxies in total. Note that although these groups of galaxies contain usual galaxies, their aggregations are anomalous due to the way the groups are constructed.

The average precision (AP) and area under the ROC curve (AUC) from 10 random repetitions are shown in Figure 5. Based on the average precision, KNN-L2, MGM, and OCSMM achieve similar results on this dataset and KNN-Renyi outperforms all other algorithms. On the other hand, the OCSMM and KNN-Renyi achieve highest AUC scores on this dataset. Moreover, it is clear that point anomaly detection using the OCSVM fails to detect group anomalies.

5.4 High Energy Particle Physics

In this section, we demonstrate our group anomaly detection algorithm in high energy particle physics, which is largely the study of fundamental particles, e.g., neutrinos, and their interactions. Essentially, all particles and their dynamics can be described by a quantum field theory called the *Standard Model*. Hence, given massive datasets from high-energy physics experiments, one is interested in discovering deviations from known Standard Model physics.

Searching for the Higgs boson, for example, has recently received much attention in particle physics and machine learning communities (see e.g., Bhat (2011), Vatanen et al. (2012) and references therein). A new physical phenomena usually manifest themselves as tiny excesses of certain types of collision events among a vast background of known physics in particle detectors.

Anomalies occur as a cluster among the background data. The background data distribution contaminated by these anomalies will therefore be different from the true background distribution. It is very difficult to detect this difference in general because the contamination can be considerably small. In this experiment, we consider similar condition as in Vatanen et al. (2012) and generate data using the standard HEP Monte Carlo generators such as PYTHIA[3]. In particular, we consider a Monte Carlo simulated events where the Higgs is produced in association with the W boson and decays into two bottom quarks.

The data vector consists of 5 variables (p_x, p_y, p_z, e, m) corresponding to different characteristics of the topology of a collision event. The variables p_x, p_y, p_z, e rep-

[2]See http://www.sdss.org for the detail of the surveys.

[3]http://home.thep.lu.se/∼torbjorn/Pythia.html

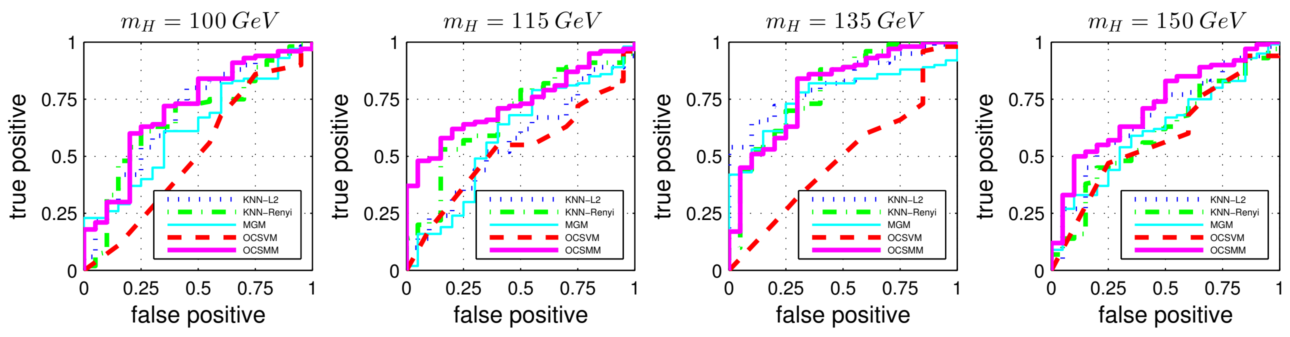


Figure 6: The ROC of different group anomaly detection algorithms on the Higgs boson datasets with various Higgs masses m_H. The associated AUC scores for different settings, sorted in the same order appeared in the figure, are (0.6835,0.6655,0.6350,0.5125,**0.7085**), (0.5645,0.6783,0.5860,0.5263,**0.7305**), (**0.8190**,0.7925,0.7630,0.4958,0.7950), and (0.6713,0.6027,0.6165,0.5862,**0.7200**).

resents the momentum four-vector in units of GeV with $c = 1$. The variable m is the particle mass in the same unit. The signal looks slightly different for different Higgs masses m_H, which is an unknown free parameter in the Standard Model. In this experiment, we consider $m_H = 100, 115, 135$, and 150 GeV. We generate 120 groups of collision events, 100 of which contain only background signals, whereas the rest also contain the Higgs boson collision events. For each group, the number of observable particles ranges from 200 to 500 particles. The goal is to detect the anomalous groups of signals which might contain the Higgs boson without prior knowledge of m_H.

Figure 6 depicts the ROC of different group anomaly detection algorithms. The OCSMM and KNN-based group anomaly detection algorithms tend to achieve competitive performance and outperform the MGM algorithm. Moreover, it is clear that traditional point anomaly detection algorithm fails to detect high-level anomalous structures.

6 Conclusions and Discussions

To conclude, we propose a simple and efficient algorithm for detecting group anomalies called one-class support measure machine (OCSMM). To handle aggregate behaviors of data points, groups are represented as probability distributions which account for higher-order information arising from those behaviors. The set of distributions are represented as mean functions in the RKHS via the kernel mean embedding. We also extend the relationship between the OCSVM and the KDE to the OCSMM in the context of variable kernel density estimation, bridging the gap between large-margin approach and kernel density estimation. We demonstrate the proposed algorithm on both synthetic and real-world datasets, which achieve competitive results compared to existing group anomaly detection techniques.

It is vital to note the differences between the OCSMM and hierarchical probabilistic models such as MGM and FGM. Firstly, the probabilistic models assume that data are generated according to some parametric distributions, i.e., mixture of Gaussian, whereas the OCSMM is nonparametric in the sense that no assumption is made about the distributions. It is therefore applicable to a wider range of applications. Secondly, the probabilistic models follow a bottom-up approach. That is, detecting group-based anomalies requires point-based anomaly detection. Thus, the performance also depends on how well anomalous points can be detected. Furthermore, it is computational expensive and may not be suitable for large-scale datasets. On the other hand, the OCSMM adopts the top-down approach by detecting the group-based anomalies directly. If one is interested in finding anomalous points, this can be done subsequently in a group-wise manner. As a result, the top-down approach is generally less computational expensive and can be used efficiently for online applications and large-scale datasets.

References

I. S. Abramson. On bandwidth variation in kernel estimates-a square root law. *The Annals of Statistics*, 10(4):1217–1223, 1982.

A. Berlinet and T. C. Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers, 2004.

P. C. Bhat. Multivariate Analysis Methods in Particle Physics. *Ann.Rev.Nucl.Part.Sci.*, 61:281–309, 2011.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3: 993–1022, 2003.

J. Bovy, J. F. Hennawi, D. W. Hogg, A. D. Myers, J. A. Kirkpatrick, D. J.Schlegel, N. P. Ross, E. S. Sheldon,

I. D. McGreer, D. P. Schneider, and B. A. Weaver. Think outside the color box: Probabilistic target selection and the sdss-xdqso quasar targeting catalog. *The Astrophysical Journal*, 729(2):141, 2011.

L. Breiman, W. Meisel, and E. Purcell. Variable kernel estimates of multivariate densities. *Technometrics*, 19 (2):135–144, 1977.

P. K. Chan and M. V. Mahoney. Modeling multiple time series for anomaly detection. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM)*, pages 90–97. IEEE Computer Society, 2005.

V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):1–58, July 2009.

K. Das, J. Schneider, and D. B. Neill. Anomaly pattern detection in categorical datasets. In *ACM-SIGKDD*, pages 169–176. ACM, 2008.

K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, December 2004.

H. Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40(3):863–874, 2007.

J. A. Kirkpatrick, D. J. Schlegel, N. P. Ross, A. D. Myers, J. F. Hennawi, E. S.Sheldon, D. P. Schneider, and B. A. Weaver. A simple likelihood method for quasar target selection. *The Astrophysical Journal*, 743(2):125, 2011.

K. Muandet, K. Fukumizu, F. Dinuzzo, and B. Schölkopf. Learning from distributions via support measure machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 10–18. 2012.

B. Póczos, L. Xiong, and J. G. Schneider. Nonparametric divergence estimation with applications to machine learning on distributions. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 599–608, 2011.

N. P. Ross, A. D. Myers, E. S. Sheldon, C. Yche, M. A. Strauss, J. Bovy, J. A. Kirkpatrick, G. T. Richards, ric Aubourg, M. R. Blanton, W. N. Brandt, W. C. Carithers, R. A. C. Croft, R. da Silva, K. Dawson, D. J. Eisenstein, J. F. Hennawi, S. Ho, D. W. Hogg, K.-G. Lee, B. Lundgren, R. G. McMahon, J. Miralda-Escud, N. Palanque-Delabrouille, I. Pris, P. Petitjean, M. M. Pieri, J. Rich, N. A. Roe, D. Schiminovich, D. J. Schlegel, D. P. Schneider, A. Slosar, N. Suzuki, J. L. Tinker, D. H. Weinberg, A. Weyant, M. White, and W. M. Wood-Vasey. The sdss-iii baryon oscillation spectroscopic survey: Quasar target selection for data release nine. *The Astrophysical Journal Supplement Series*, 199 (1):3, 2012.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001. ISBN 0262194759.

B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13:1443–1471, 2001.

A. Smola, A. Gretton, L. Song, and B. Schölkopf. A hilbert space embedding for distributions. In *Proceedings of the 18th International Conference on Algorithmic Learning Theory*, pages 13–31. Springer-Verlag, 2007.

B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 2010.

D. M. J. Tax and R. P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20:1191–1199, 1999.

D. M. J. Tax and R. P. W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

G. R. Terrell and D. W. Scott. Variable kernel density estimation. *The Annals of Statistics*, 20(3):1236–1265, 1992.

T. Vatanen, M. Kuusela, E. Malmi, T. Raiko, T. Aaltonen, and Y. Nagai. Semi-supervised detection of collective anomalies with an application in high energy particle physics. In *IJCNN*, pages 1–8. IEEE, 2012.

L. Xiong, B. Poczos, and J. Schneider. Group anomaly detection using flexible genre models. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2011a.

L. Xiong, B. Póczos, J. G. Schneider, A. Connolly, and J. VanderPlas. Hierarchical probabilistic models for group anomaly detection. *Journal of Machine Learning Research - Proceedings Track*, 15:789–797, 2011b.

M. Zhao and V. Saligrama. Anomaly detection with score functions based on nearest neighbor graphs. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 2250–2258, 2009.

Structured Convex Optimization under Submodular Constraints

Kiyohito Nagano
Dept. of Complex and Intelligent Systems
Future University Hakodate
k_nagano@fun.ac.jp

Yoshinobu Kawahara
The Institute of Scientific and Industrial Research
Osaka University
kawahara@ar.sanken.osaka-u.ac.jp

Abstract

A number of discrete and continuous optimization problems in machine learning are related to convex minimization problems under submodular constraints. In this paper, we deal with a submodular function with a directed graph structure, and we show that a wide range of convex optimization problems under submodular constraints can be solved much more efficiently than general submodular optimization methods by a reduction to a maximum flow problem. Furthermore, we give some applications, including sparse optimization methods, in which the proposed methods are effective. Additionally, we evaluate the performance of the proposed method through computational experiments.

1 Introduction

A submodular function is a fundamental tool in discrete optimization, machine learning and other related fields and has been recognized as an interesting subject of research. A submodular function is known to be a discrete counterpart of a convex function (Lovász [17]). Especially, the *submodular function minimization* problem is an elemental problem, and many combinatorial problems arising in machine learning, such as clustering [25, 24], image segmentation [31] and feature selection [2], can be reduced to this problem.

For example, Narasimhan, Joic and Bilmes [25] showed that clustering problems with some specific natural criteria, such as the minimum description length, can be solved as the problem of minimizing a symmetric submodular function. Also, Bach [2] recently showed that many of the known structured-sparsity inducing norms can be interpreted as continuous relaxations, called the Lovász extensions, of submodular functions. Based on this correspondence relationship, proximal operators, which are required for learning with structured regularization, can be computed as minimum-norm-point problems on submodular polyhedra.

Similarly to convex functions, submodular functions can be exactly minimized in polynomial time. The fastest known algorithm of Orlin [27] runs in $\mathrm{O}(n^5\mathrm{EO}+n^6)$ time, where n is the size of the ground set and EO is the time for function evaluation. On the other hand, the minimum norm point algorithm (Fujishige [9]) is usually much faster in practice [10], although it has worse time complexity. However, the existing algorithms for the general submodular minimization problem, even including the minimum norm point algorithm, do not scale sufficiently to large problems from a practical point of view.

Meanwhile, it is known that submodular function minimization problems can be solved more efficiently when the submodular functions have particular structure. For symmetric submodular functions, Queyranne [29] gave a minimization algorithm that runs in $\mathrm{O}(n^3\mathrm{EO})$. Also recently, Stobbe and Krause [31] introduced a decomposable submodular function and developed the Smoothed Lovász Gradient (SLG) algorithm, which is based on the smoothing technique of Nesterov [26] and the discrete convexity of a submodular function. In addition, Jegelka et al. [14] introduced a generalized graph cut function, which generalizes a large subfamily of submodular functions, and proposed an efficient network flow based minimization algorithm.

In this paper, we consider a separable convex optimization problem over a base polyhedron, which is a discrete structure determined by a submodular function. Separable convex optimization under submodular constraints is related to various discrete and continuous optimization problems, including network analysis methods [23], sparse learning methods [2], and approximation algorithms for NP-hard combinatorial optimization problems [13]. For a general submodular function, separable convex optimization problems can be solved within the same running time as submodular function minimization [6, 21], that is, $\mathrm{O}(n^5\mathrm{EO} + n^6)$ time. Thus, such algorithms are impractical when the size of the ground set is large. Even though the minimum norm point algorithm [9] and its weighted version [22] would solve such quadratic minimization problem much faster, it does not have good time complexity bounds and still does not scale to large problems.

We show that if a submodular function has a specific graph structure, the convex optimization problem can be solved efficiently with the aid of a general framework of the decomposition algorithm [9, 22] and network flow algorithms [11, 12, 28]. We develop a parametrized directed graph structure that determines a parametric submodular function minimization problem, and show that the decomposition algorithm can be performed successfully by computing the maximal minimum cuts iteratively. Furthermore, we mention that several machine learning applications can be solved in this convex optimization problem. We remark that the proposed method can deal with a relatively general submodular function and various separable convex objective functions.

The remainder of the paper is organized as follows. In Section 2, we provide the definitions of basic concepts and give a definition of a convex optimization problem under submodular constraints. In Section 3, we give examples of submodular functions that have good graph structures. In Section 4, we show some optimization problems related to separable convex optimization problems under submodular constraints. In Section 5, we describe a general decomposition algorithm for solving separable convex optimization problems, and in Section 6, we further show that structured convex optimization problems under submodular constraints can be solved efficiently with the aid of network flow algorithms. Finally, we show some empirical results of computational experiments in Section 7, and give concluding remarks in Section 8.

2 Submodular functions and convex optimization problems

We give basic definitions of a submodular function and related concepts (for details on the theory of submodular functions, see [9, 30]). Then, we give the definition of a convex optimization problem under submodular constraints.

2.1 Submodular functions and related polyhedra

Let $V = \{1, \dots, n\}$ be a given set of n elements, and let $g : 2^V \to \mathbb{R}$ be a real-valued function defined on all the subset of V. Such a function g is called a *set function* with a ground set V. The set function $g : 2^V \to \mathbb{R}$ is called *submodular* if

$$g(S) + g(T) \geq g(S \cup T) + g(S \cap T), \forall S, T \subseteq V. \quad (1)$$

A set function g is called *supermodular* if $-g$ is submodular. A set function is called *modular* if it always satisfies (1) with equality. A set function is called *nondecreasing* if $g(S) \leq g(T)$ for any $S, T \subseteq V$ with $S \subseteq T$. For an n-dimensional vector $\boldsymbol{a} \in \mathbb{R}^n$ with components a_i, $i \in V$, and a subset $S \subseteq V$, we denote $a(S) = \sum_{i \in S} a_i$. For convenience, we let $a(\varnothing) = 0$. A set function $a : 2^V \to \mathbb{R}$ corresponding to the vector $\boldsymbol{a}$ is a modular function.

Submodular function minimization

A submodular function minimization problem is a fundamental unifying discrete optimization problem. For a submodular function $g : 2^V \to \mathbb{R}$, the submodular function minimization problem asks for finding a subset $S \subseteq V$ that minimizes $f(S)$. This problem is known to be solvable in polynomial time, and the fastest known polynomial time algorithm [27] that runs in $\mathrm{O}(n^5\mathrm{EO} + n^6)$ time, where EO is the time of one function evaluation of g. The algorithms for general submodular function minimization are impractical when $n = |V|$ is large. In addition, the minimum norm point algorithm [9] is known to be usually much faster in practice, although it has worse time complexity.

Let $\mathrm{Arg\,min}\, g \subseteq 2^V$ denote the family of all minimizers of g. That is, $\mathrm{Arg\,min}\, g = \{S^* \subseteq V : f(S^*) = \min_S f(S)\}$. For S^*, $T^* \in \mathrm{Arg\,min}\, g$, the submodularity of g implies that $S^* \cup T^*$, $S^* \cap T^* \in \mathrm{Arg\,min}\, g$. Thus, there exist the (unique) minimal minimizer and the (unique) maximal minimizer of g. Many submodular function minimization algorithms can be modified to find the maximal minimizer and/or the minimal minimizer (see, e.g., [21]).

Base polyhedron

For a submodular function $g : 2^V \to \mathbb{R}$ with $g(\varnothing) = 0$, the *submodular polyhedron* $\mathrm{P}(g) \subseteq \mathbb{R}^n$ and the *base polyhedron* $\mathrm{B}(g) \subseteq \mathbb{R}^n$ are given by

$$\begin{aligned}\mathrm{P}(g) &= \{\boldsymbol{x} \in \mathbb{R}^n : x(S) \leq g(S) \ (\forall S \subseteq V)\},\\ \mathrm{B}(g) &= \{\boldsymbol{x} \in \mathrm{P}(g) : x(V) = g(V)\}.\end{aligned}$$

Figure 1 illustrates examples of the base polyhedra. $\mathrm{B}(g)$ is determined by 2^n-2 inequalities and one equality. We see that $\mathrm{B}(g)$ is nonempty and bounded. The base polyhedron $\mathrm{B}(g)$ is included in the nonnegative orthant $\mathbb{R}^n_{\geq 0}$ if and only if g is nondecreasing.

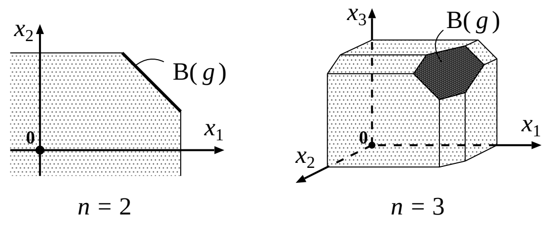


Figure 1: Examples of base polyhedra

2.2 Convex optimization under submodular constraints

Throughout this paper, we suppose that set function $f : 2^V \to \mathbb{R}$ is submodular and satisfies $f(\varnothing) = 0$. Let $w_i : \mathbb{R} \to \mathbb{R}$ be a convex function for each $i \in V$. We consider the separable convex function minimization problem over the base polyhedron :

$$\min_{\boldsymbol{x}\in \mathrm{B}(f)} \sum_{i\in V} w_i(x_i). \tag{2}$$

It is known that a number of optimization problems of this form are equivalent.

Theorem 1 (Nagano and Aihara [22]). *Suppose that $f : 2^V \to \mathbb{R}$ is a nondecreasing submodular function with $f(\varnothing) = 0$. Let $\boldsymbol{b} \in \mathbb{R}^n$ be a positive vector, and let $w_0 : \mathbb{R} \to \mathbb{R}$ be a differentiable and strictly convex function. The following problems* (1.a) – (1.f) *have the same (optimal) solution:*

problem (1.a) $\min_{\boldsymbol{x}\in \mathrm{B}(f)} \sum_{i=1}^{n} \frac{x_i^2}{b_i}$;

problem (1.b) $\min_{\boldsymbol{x}\in \mathrm{B}(f)} \sum_{i=1}^{n} \frac{x_i^{p+1}}{b_i^p}$ *for* $p>0$;

problem (1.c) $\max_{\boldsymbol{x}\in \mathrm{B}(f)} \sum_{i=1}^{n} \frac{x_i^{p+1}}{b_i^p}$ *for* $p<0$ *with* $p\neq -1$;

problem (1.d) $\max_{\boldsymbol{x}\in \mathrm{B}(f)} \sum_{i=1}^{n} b_i \ln x_i$;

problem (1.e) $\min_{\boldsymbol{x}\in \mathrm{B}(f)} \sum_{i=1}^{n} (x_i \ln \frac{x_i}{b_i} + b_i - x_i)$;

problem (1.f) $\min_{\boldsymbol{x}\in \mathrm{B}(f)} \sum_{i=1}^{n} x_i g_0(\frac{b_i}{x_i})$.

In view of Theorem 1, we focus on the case where the objective function is quadratic. For a positive vector $\boldsymbol{b} \in \mathbb{R}^n$, we mainly deal with problem (1.a). By using the following two observations, w.l.o.g., we can assume that the submodular function f is nondecreasing.

Lemma 2. *For any $\beta \in \mathbb{R}$, $\boldsymbol{x}^*$ is optimal for $\min\{\sum_i \frac{x_i^2}{b_i} : x \in \mathrm{B}(f)\}$ if and only if $\boldsymbol{x}^* + \beta\boldsymbol{b}$ is optimal for $\min\{\sum_i \frac{x_i^2}{b_i} : x \in \mathrm{B}(f + \beta b)\}$.*

Lemma 3. *Set $\beta := \max\{0, \max_{i=1,\dots,n} \frac{f(V\setminus\{i\})-f(V)}{b_i}\}$. Then $f + \beta b$ is a nondecreasing submodular function.*

Problem (1.a) is known as the lexicographically optimal base problem [8]. If $\boldsymbol{b}$ is the all-one vector, problem (1.a) becomes the minimum norm base problem. For a general submodular function, problem (1.a) can be solved within the same running time as the submodular function minimization [6, 21], that is, $\mathrm{O}(n^5\mathrm{EO}+n^6)$ time, where EO is the time of one function evaluation. Thus, such algorithms are impractical when $n = |V|$ is large. Although the minimum norm point algorithm [9] and its weighted version [22] can solve problem (1.a) much faster, it has worse time complexity and still does not scale to large problems.

In this paper, we point out that if the function f has a good graph structure, problem (1.a) can be solved efficiently with the aid of network flow algorithms. Furthermore, we show a number of applications of the convex optimization problem (1.a).

3 Structured submodular functions and minimization problems

Many basic submodular functions can be represented by using graphs. In such cases, a minimum cut algorithm, which runs much faster in practice, is useful to solve submodular optimization problems.

In this section, we will see some examples of submodular functions with directed graph structures, which are important from the viewpoint of applications.

3.1 Minimizing graph cut functions

In this subsection, we will see that an s-t cut function $\kappa_{s\text{-}t}$ and a generalized graph cut function γ of [14], both of which are submodular, can be minimized efficiently with the aid of network flow algorithms. In particular, we will see that the maximal minimizer can be computed efficiently in both cases. In the general algorithm described in Section 5, the maximal minimizer of a submodular function has to be computed.

Minimum cut problem

We start with the minimum s-t cut problem. Let $\mathcal{G} = (\{s\} \cup \{t\} \cup \mathcal{V}, \mathcal{E})$ be a directed graph, where s is a special source node, t is a special sink node, $\mathcal{V}$ is a set of other nodes, and $\mathcal{E}$ is a set of directed edges. For each $e \in \mathcal{E}$, a nonnegative capacity value $c(e)$ is assigned. An s-t cut is an ordered bipartition $(\mathcal{V}_1, \mathcal{V}_2)$ of the node set of $\mathcal{G}$ such that $s \in \mathcal{V}_1$ and $t \in \mathcal{V}_2$. Clearly, any s-t cut can be expressed as $(\{s\}\cup S, \{t\}\cup(\mathcal{V}\setminus S))$ for some $S \subseteq \mathcal{V}$. For an s-t cut $(\{s\}\cup S, \{t\}\cup(\mathcal{V}\setminus S))$, its capacity $\kappa_{s\text{-}t}$ is defined by

$$\kappa_{s\text{-}t}(S) = \sum\{c(e) : e \in \delta_{\mathcal{G}}^{\mathrm{out}}(\{s\}\cup S)\} \tag{3}$$

for each $S \subseteq \mathcal{V}$, where $\delta^{\mathrm{out}}_{\mathcal{G}}(\mathcal{V}')$ is a set of edges leaving $\mathcal{V}' \subseteq \mathcal{V}$ in $\mathcal{G}$. The minimum cut problem asks for finding an s-t cut of $\mathcal{G}$ that minimizes the capacity. The set function $\kappa_{s\text{-}t} : 2^{\mathcal{V}} \to \mathbb{R}$, which is called an *$s$-$t$ cut function*, is known to be submodular. Therefore, the minimum cut problem is a special case of a submodular function minimization problem.

The minimum cut problem is closely related to the maximum flow problem, which is a fundamental problem in combinatorial optimization [1]. It can be solved quite efficiently. For example, it can be solved in $\mathrm{O}(|\mathcal{V}||\mathcal{E}| \log(|\mathcal{V}|^2/|\mathcal{E}|))$ time [12] or $\mathrm{O}(|\mathcal{V}||\mathcal{E}|)$ time [28].

As $\kappa_{s\text{-}t}$ is submodular, there exists the maximal minimizer $S^*_{\max}$ of $\kappa_{s\text{-}t}$. The s-t cut $(\{s\} \cup S^*_{\max}, \{t\} \cup (\mathcal{V} \setminus S^*_{\max}))$ is called the *maximal minimum s-t cut*. Once a maximal flow is computed, we can obtain the maximal minimum s-t cut in additional $\mathrm{O}(|\mathcal{V}| + |\mathcal{E}|)$ time (we just need to consider the set of nodes reachable to the sink t and its complement in the residual network [1]). The minimal minimum s-t cut can be defined and computed in a similar way.

Lemma 4. *The maximal minimizer of the s-t cut function $\kappa_{s\text{-}t} : 2^{\mathcal{V}} \to \mathbb{R}$ defined in* (3) *can be computed in* $\mathrm{O}(|\mathcal{V}||\mathcal{E}| \log(|\mathcal{V}|^2/|\mathcal{E}|))$ *time, or,* $\mathrm{O}(|\mathcal{V}||\mathcal{E}|)$ *time.*

Generalized graph cut functions

Next we give a definition of the generalized graph cut function $\gamma : 2^V \to \mathbb{R}$ of Jegelka et al. [14], which generalizes a large subfamily of submodular functions.

Let $\mathcal{G} = (\{s\} \cup \{t\} \cup \mathcal{V}, \mathcal{E})$ be a directed graph with nonnegative edge capacities $c(e) \geq 0$ $(e \in \mathcal{E})$. Suppose that the set $\mathcal{V}$ is partitioned as $\mathcal{V} = V \cup U$, where $V = \{1, \ldots, n\}$ is a set of nodes, each of which may become a source, and U is a set of auxiliary nodes (U can be empty). Figure 2 illustrates an example of the graph $\mathcal{G} = (\{s\} \cup \{t\} \cup V \cup U, \mathcal{E})$. A generalized graph cut function [14] $\gamma : 2^V \to \mathbb{R}$ is defined by

$$\gamma(S) = \min_{W \subseteq U} \sum\{c(e) : e \in \delta^{\mathrm{out}}_{\mathcal{G}}(\{s\} \cup S \cup W)\} \qquad (4)$$

for each $S \subseteq V$. If U is empty, the function γ coincides with the function $\kappa_{s\text{-}t}$ defined in (3). The submodularity of γ can be derived from the classical result of Megiddo [20] on network flow problems with multiple terminals (for details, see the appendix of this paper).

Let us consider the minimization of $\gamma : 2^V \to \mathbb{R}$. By the definition of γ, the value $\gamma^* := \min_{S \subseteq V} \gamma(S)$ is equal to the capacity of a minimum s-t cut in $\mathcal{G}$. For any minimum s-t cut $(\{s\} \cup \mathcal{P}, \{t\} \cup (V \cup U \setminus \mathcal{P}))$ in $\mathcal{G}$, we have $\gamma(\mathcal{P} \cap V) = \gamma^*$ and thus $\mathcal{P} \cap V$ is a minimizer of $\gamma : 2^V \to \mathbb{R}$. Therefore, a minimizer of γ can be computed by solving the minimum s-t cut problem on $\mathcal{G} = (\{s\} \cup \{t\} \cup V \cup U, \mathcal{E})$.

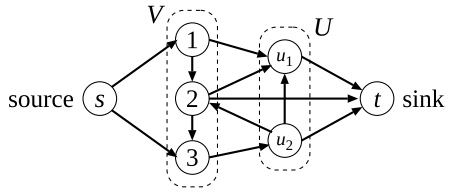


Figure 2: A directed graph $\mathcal{G} = (\{s\} \cup \{t\} \cup V \cup U, \mathcal{E})$ that generates a generalized graph cut function $\gamma : 2^V \to \mathbb{R}$

Conversely, let $S^* \subseteq V$ be a minimizer of γ, and let W^* be a subset $W \subseteq U$ that attains the minimum in the right hand side of (4) with respect to $S = S^*$. Then $S^* \cup W^* \subseteq \mathcal{V}$ minimizes the s-t cut function (see [14]).

Therefore, given the maximal minimum s-t cut $(\{s\} \cup \mathcal{P}^*_{\max}, \{t\} \cup (V \cup U \setminus \mathcal{P}^*_{\max}))$, the subset $\mathcal{P}^*_{\max} \cap V$ is the maximal minimizer of γ.

Lemma 5. *The maximal minimizer of the generalized graph cut function $\gamma : 2^V \to \mathbb{R}$ defined in* (4) *can be computed in* $\mathrm{O}(|\mathcal{V}||\mathcal{E}| \log(|\mathcal{V}|^2/|\mathcal{E}|))$ *time, or,* $\mathrm{O}(|\mathcal{V}||\mathcal{E}|)$ *time, where $\mathcal{V} = V \cup U$.*

3.2 Transformed graph cut functions

We define a transformed graph cut function, and we show that the function can be regarded as an s-t cut function defined in Subsection 3.1. In Subsection 4.1, we will see that the convex minimization problem (1.a) under the constraints of this function is related to the densest subgraph problem.

Let $G = (V, E)$ be a directed graph with node set $V = \{1, \ldots, n\}$ and edge set E. Given nonnegative edge capacities $c(e)$ $(e \in E)$, a cut function $\kappa : 2^V \to \mathbb{R}$ defined by $\kappa(S) = \sum\{c(e) : e \in \delta^{\mathrm{out}}_G(S)\}$ for each $S \subseteq V$ is submodular. Let $\boldsymbol{a} \in \mathbb{R}^n$. Then, a *transformed graph cut function* $\kappa_a : 2^V \to \mathbb{R}$ defined by

$$\kappa_a = \kappa + a$$

is also submodular.

Let us see that the function $\kappa_a : 2^V \to \mathbb{R}$ can be regarded as an s-t cut function on a new graph $\mathcal{G}_{\boldsymbol{a}}$.

Define $A_+ = \{i \in V : a_i > 0\}$ and $A_- = \{i \in V : a_i < 0\}$. By adding new nodes s, t and new edges $E_+ \cup E_-$ to G, we construct a new directed graph $\mathcal{G}_{\boldsymbol{a}} = (\{s\} \cup \{t\} \cup V, E \cup E_+ \cup E_-)$, where $E_+ = \{(i, t) : i \in A_+\}$ and $E_- = \{(s, i) : i \in A_-\}$. The capacities of new edges are determined as follows: we set $c(i, t) = a_i$ (≥ 0) for each $(i, t) \in E_+$, and set $c(s, i) = -a_i$ (≥ 0) for each $(s, i) \in E_-$. Figure 3 shows the construction of $\mathcal{G}_{\boldsymbol{a}}$.

For an s-t cut $(\{s\} \cup S, \{t\} \cup (V \setminus S))$ of $\mathcal{G}_{\boldsymbol{a}}$, its capacity

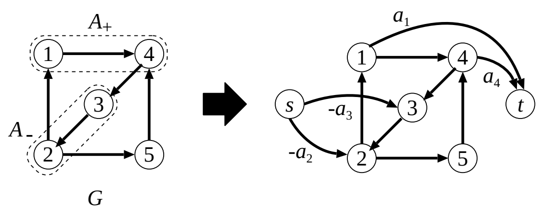


Figure 3: Construction of the directed graph $\mathcal{G}_{\boldsymbol{a}}$

is equal to

$$\begin{aligned}&\kappa(S) + a(S \cap A_+) + (-a(A_- \setminus S))\\ &= \kappa(S) + a(S) - a(A_-)\\ &= \kappa_a(S) + \text{const.}\end{aligned}$$

Thus, κ_a can be regarded as an s-t cut function on $\mathcal{G}_{\boldsymbol{a}}$.

3.3 Decomposable submodular functions

Decomposable submodular function (see [31]) are one of the most important special case of generalized graph cut functions [14]. For more examples of generalized graph cut functions, refer to Jegelka et al. [14].

A *decomposable submodular function* $\tau : 2^V \to \mathbb{R}$ is a set function that can be represented as a sum of a modular set function and submodular set functions arising from concave functions. As to Stobbe and Krause [31], we will focus on the case where each concave function is a threshold potential. That is, we consider the following decomposable submodular function $\tau : 2^V \to \mathbb{R}$ defined by

$$\tau(S) = -d(S) + \sum_{j=1}^{k} \min\{y_j,\, w^j(S)\} \tag{5}$$

for each $S \subseteq V$, where $\boldsymbol{d} \in \mathbb{R}^n$ is a positive vector, $\boldsymbol{w}^1, \ldots, \boldsymbol{w}^k \in \mathbb{R}^n$ are nonnegative vectors, and $y_1, \ldots, y_k > 0$.

Now we observe that the function τ defined in (5) can be represented as a generalized graph cut function defined in Subsection 3.1. Consider a directed graph $\mathcal{G}_\tau = (\{s\} \cup \{t\} \cup V \cup U, \mathcal{E})$, where $V = \{1, \ldots, n\}$, $U = \{u_1, \ldots, u_k\}$, and $\mathcal{E} = \{(s, i) : i \in V\} \cup \{(i, u_j) : i \in V,\, u_j \in U\} \cup \{(u_j, t) : u_j \in U\}$. The edge capacities are determined as

$$\begin{aligned} c(s, i) &= d_i, \ \forall i \in V,\\ c(i, u_j) &= w_i^j, \ \forall i \in V,\ \forall u_j \in U,\\ c(u_j, t) &= y_j, \ \forall u_j \in U.\end{aligned}$$

Figure 4 illustrates the directed graph $\mathcal{G}_\tau$. We can observe that $\mathcal{G}_\tau$ generates the decomposable submodular function τ.

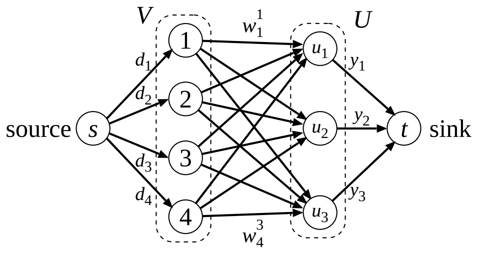


Figure 4: Directed graph $\mathcal{G}_\tau$ associated with a decomposable submodular function τ with $n = 4$ and $k = 3$

The function τ corresponds to the sum of truncated functions described in [14], and the construction of $\mathcal{G}_\tau$ is widely used in computer vision [15].

4 Applications

It is known that the convex optimization problem (2) under submodular constraints is related to some discrete and continuous optimization problems. In this section, we show some examples in which the submodular functions have graph structures considered in Section 3.

4.1 Finding dense subgraphs

Let $\overline{G} = (V, \overline{E})$ be an undirected graph with node set $V = \{1, \ldots, n\}$ and undirected edge set $\overline{E}$. Given nonnegative edge capacities $c(e)$ $(e \in \overline{E})$ and an integer k, the densest k-subgraph problem asks for finding a k-subset $S \subseteq V$ that maximizes $\theta(S)$, where $\theta(S)$ is the sum of weights of edges in the subgraph induced by S. The function $\theta : 2^V \to \mathbb{R}$ is a supermodular function with $\theta(\varnothing) = 0$, and the minimum norm base problem

$$\min_{\boldsymbol{x} \in \mathbf{B}(-\boldsymbol{\theta})} \sum_{i=1}^{n} x_i^2 \tag{6}$$

plays an important role to find dense subgraphs of $\overline{G}$ [23].

We show that $-\theta$ is a transformed graph cut function (Subsection 3.2). Let $\boldsymbol{m} \in \mathbb{R}^n$ be a vector defined by $m_i = \sum_{i'}\{c(\{i, i'\}) : \{i, i'\} \in \overline{E}\}$ for each $i \in V$, and let $\overline{\kappa}$ be a cut function of $\overline{G}$, that is, $\overline{\kappa}(S) = \sum\{c(\{i, i'\}) : \{i, i'\} \in \overline{E}, i \text{ in } S \text{ and } i' \text{ in } V \setminus S\}$ $(S \subseteq V)$. Then we have

$$-\theta(S) = \tfrac{1}{2}\overline{\kappa}(S) - \tfrac{1}{2}m(S)$$

for each $S \subseteq V$. It is easy to see that the function $\overline{\kappa} : 2^V \to \mathbb{R}$ can be regarded as a cut function of a directed graph. Thus, $-\theta$ is a transformed graph cut function.

4.2 Proximal methods

Regularized learning is a fundamental formulation for many supervised problems. Let $\{(\boldsymbol{z}_i, y_i)\}_{i=1}^N$ be a set of samples, $\boldsymbol{\beta} \in \mathbb{R}^n$ a model parameter vector and $l(\boldsymbol{z}, y; \boldsymbol{\beta})$ a (differentiable) convex loss. Then, the optimization for regularized learning is represented as

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^n} \sum_{i=1}^N l(\boldsymbol{z}_i, y_i; \boldsymbol{\beta}) + \lambda \cdot \Omega(\boldsymbol{\beta}),$$

where $\Omega(\boldsymbol{\beta})$ is a regularization term and λ is the regularization parameter. If $\Omega(\boldsymbol{\beta})$ is non-differentiable on $\boldsymbol{\beta}$, which is usually true for structured regularization, the proximal method is a popular approach to solve this optimization problem [3]. As is well known, its update procedure at each iteration can be reduced to the calculation of the following problem:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^n} \frac{1}{2}\|\boldsymbol{\beta} - \boldsymbol{s}\|_2^2 + \lambda \cdot \Omega(\boldsymbol{\beta}), \tag{7}$$

where $\boldsymbol{s} \in \mathbb{R}^n$. Recently, Bach [2] showed that many of the popular structured norms can be represented as continuous relaxations, called Lovász extensions, of submodular functions. And in this case, Problem (7) can be transformed to

$$\min_{\boldsymbol{t}} \{\textstyle\sum_{i=1}^n t_i^2 : \boldsymbol{t} \in \mathrm{B}(g - \lambda^{-1} s)\},$$

where g is a submodular function whose Lovász extension is Ω ($\boldsymbol{\beta}$ is solved as $\lambda \boldsymbol{t}$). Note that many of popular structured norms can be expressed as the Lovász extensions of generalized graph cut functions, such as cut functions (that correspond to fused-regularization)[4] and coverage functions (that correspond to overlapping group-regularization).

4.3 Minimum ratio problems

For a nonnegative submodular function $g : 2^V \to \mathbb{R}$ with $g(\varnothing) = 0$ and a positive vector $\boldsymbol{b} \in \mathbb{R}^n$, consider the minimum ratio problem which asks for a subset $S \in 2^V \setminus \{\varnothing\}$ minimizing $g(S)/b(S)$. This kind of optimization problems have to be solved iteratively, e. g., in the primal-dual approximation algorithm for a submodular cost covering problem [13].

Suppose that we have the optimal solution $\boldsymbol{x}^*$ to $\min\{\sum_{i=1}^n \frac{x_i^2}{b_i} : \boldsymbol{x} \in \mathrm{B}(f)\}$. Let $\xi_1 = \min_{i \in V} \frac{x_i^*}{b_i}$ and let $S_1 = \{i \in V : \frac{x_i^*}{b_i} = \xi_1\}$. Then the subset S_1 is an optimal solution to the minimum ratio problem (see [9]). Therefore, by solving the separable quadratic minimization problem over $\mathrm{B}(g)$, an optimal solution to the minimum ratio problem can be obtained. If the function g has a graph structure, the running time of the approximation algorithm of [13] could be improved.

5 A general framework for separable convex minimization under submodular constraints

In this section, we describe the decomposition algorithm, which is a general framework to solve the separable convex minimization problem under submodular constraints. Before describing the decomposition algorithm, we give a parametric formulation of problem (1.a).

For the validity of the decomposition algorithm described here, e. g., refer to Fujishige [9], and Nagano and Aihara [22].

5.1 A parametric formulation

Let $f : 2^V \to \mathbb{R}$ be a general nondecreasing submodular function and $\boldsymbol{b} \in \mathbb{R}^n$ a positive vector. Recall that the set function b associated to $\boldsymbol{b}$ is modular.

For a parameter $\alpha \geq 0$, define $f_\alpha : 2^V \to \mathbb{R}$ by $f_\alpha = f - \alpha b$, which is submodular. Let us see how problem (1.a) can be reduced to the parametric submodular minimization problem: minimize f_α for all $\alpha \geq 0$. It is known that there exist $\ell + 1$ subsets,

$$(\varnothing =)\ S_0 \subset S_1 \subset \cdots \subset S_\ell\ (= V),$$

and $\ell + 1$ subintervals of $\mathbb{R}_{\geq 0} = \{\alpha \in \mathbb{R} : \alpha \geq 0\}$,

$$R_0 = [0, \alpha_1),\ R_1 = [\alpha_1, \alpha_2),\ \ldots,$$
$$R_j = [\alpha_j, \alpha_{j+1}),\ \ldots,\ R_\ell = [\alpha_\ell, +\infty),$$

such that, for each $j \in \{0, \ldots, \ell\}$, the subset S_j is the unique maximal minimizer of $f_\alpha = f - \alpha b$ for all $\alpha \in R_j$. The vector $\boldsymbol{x}^* \in \mathbb{R}^n$ determined by, for each $i \in V$ with $i \in S_{j+1} \setminus S_j$ ($j \in \{1, \ldots, \ell\}$),

$$x_i^* = \tfrac{f(S_{j+1}) - f(S_j)}{b(S_{j+1} \setminus S_j)} b_i \tag{8}$$

is the unique optimal solution to the quadratic minimization problem (1.a). The equation (8) implies that problem (1.a) can be immediately solved if the collection $\mathcal{S}^* = \{S_0, S_1, \ldots, S_\ell\}$ is computed.

5.2 The decomposition algorithm

By successively minimizing $f_\alpha = f - \alpha b$ for some appropriately chosen $\alpha \geq 0$, the decomposition algorithm finds S_j one by one, and finally the chain $S_0 \subset S_1 \subset \cdots \subset S_\ell$ and the optimal solution $\boldsymbol{x}^*$ to problem (1.a) are obtained.

The decomposition algorithm DA is recursive. Suppose that we are given two subsets $S_j, S_{j'} \in \mathcal{S}^*$ with $0 \leq j < j' \leq n$. The algorithm $\mathsf{DA}(S_j, S_{j'})$ finds the collection

$$\mathcal{S}^*(S_j, S_{j'}) := \{S \in \mathcal{S}^* : S_j \subseteq S \subseteq S_{j'}\}.$$

It can be verified that $\alpha_{j+1} \leq \frac{f(S_{j'})-f(S_j)}{b(S_{j'}\setminus S_j)} \leq \alpha_{j'}$. Therefore, we can decide if $(j+1=j')$ or $(j+1<j')$ by minimizing f_α with $\alpha = \frac{f(S_{j'})-f(S_j)}{b(S_{j'}\setminus S_j)}$.

The decomposition algorithm DA can be described as follows (see, e.g., [22] for the detailed analysis of the algorithm).

```
Algorithm DA($T$, $T'$)
  Input:   Subsets $T, T' \in \mathcal{S}^*$ with $T \subset T'$.
  Output:  The collection $\mathcal{S}^*(T, T')$.

1: Set $\alpha = \frac{f(T')-f(T)}{b(T'\setminus T)}$. Compute the unique maximal minimizer $T''$ of $f_\alpha := f - \alpha b$.
2: If $T'' = T'$, return $\{T, T'\}$.
3: If $T \subset T'' \subset T'$, let $\mathcal{S}_1$ and $\mathcal{S}_2$ be the collections returned by DA($T, T''$) and DA($T'', T'$), respectively. Return $\mathcal{S}_1 \cup \mathcal{S}_2$.
```

First of all, we know that $S_0 = \varnothing$ and $S_\ell = V$, although we do not know how large ℓ is. Clearly, we have $\mathcal{S}^*(\varnothing, V) = \mathcal{S}^*$. Therefore, by performing DA($\varnothing$, V), the collection $\mathcal{S}^*$ can be obtained. Using (8), we can immediately obtain the optimal solution of problem (1.a).

In the decomposition algorithm DA($\varnothing$, V), we minimize the functions $f_\alpha : 2^V \to \mathbb{R}$ at most $2n-1$ times.

6 Efficient algorithms for structured convex minimization problems

Let $\gamma : 2^V \to \mathbb{R}$ be a generalized graph cut function defined as in (4), which is generated from a directed graph $\mathcal{G} = (\{s\} \cup \{t\} \cup V \cup U, \mathcal{E})$. Consider the convex optimization problem (1.a) with $f = \gamma$,

$$\min_{\boldsymbol{x} \in \mathrm{B}(\gamma)} \sum_{i=1}^{n} \frac{x_i^2}{b_i}. \tag{9}$$

Recall that $\boldsymbol{b} \in \mathbb{R}^n$ is a positive vector. We show that problem (9) can be solved efficiently using the framework of the decomposition algorithm DA of Section 5.

For nonnegative parameters α and β, let us see that the set functions $\gamma - \alpha b$ and $\gamma + \beta b$ are both generalized graph cut functions. By adding new edges $e_i^- = (s, i)$ $(i \in V)$ with edge capacities $c(e_i^-) = \alpha b_i$ $(i \in V)$ to $\mathcal{G}$, we construct a new directed graph $\mathcal{G}^-_{\alpha\boldsymbol{b}}$ (see Figure 5 (a)). By adding new edges $e_i^+ = (i, t)$ $(i \in V)$ with edge capacities $c(e_i^+) = \beta b_i$ $(i \in V)$ to $\mathcal{G}$, we construct a new directed graph $\mathcal{G}^+_{\beta\boldsymbol{b}}$ (see Figure 5 (b)). Since γ is defined as in (4), the functions $\gamma - \alpha b$ and $\gamma + \beta b$ are generated by $\mathcal{G}^-_{\alpha\boldsymbol{b}}$ and $\mathcal{G}^+_{\beta\boldsymbol{b}}$, respectively.

Using Lemmas 2 and 3, and the fact that $\gamma + \beta b$ is a generalized graph cut function, we can assume that γ is nondecreasing in problem (9).

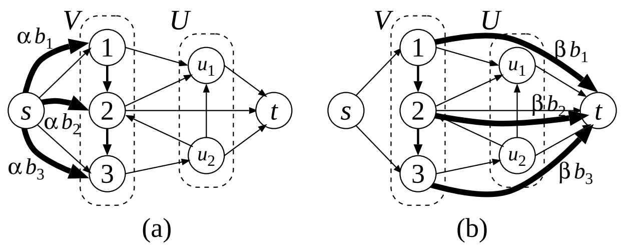


Figure 5: Directed graphs $\mathcal{G}^-_{\alpha\boldsymbol{b}}$ and $\mathcal{G}^+_{\beta\boldsymbol{b}}$

Now we can apply the decomposition algorithm DA($\varnothing$, V) to problem (9). In step 1 of the algorithm DA, we just have to compute the maximal minimum s-t cut in $\mathcal{G}^-_{\alpha\boldsymbol{b}}$ for some appropriately chosen $\alpha \geq 0$ to find the maximal minimizer of $\gamma - \alpha b$. Since we minimize the functions $\gamma - \alpha b$ at most $2n-1$ times, we obtain the following theorem with the aid of the minimum s-t cut algorithm [28].

Theorem 6. *For a generalized graph cut function* $\gamma : 2^V \to \mathbb{R}$ *generated from* $\mathcal{G} = (\{s\} \cup \{t\} \cup V \cup U, \mathcal{E})$, *problem* (9) *can be solved in* $\mathrm{O}(n(n + |U|)|\mathcal{E}|)$ *time, where* $n = |V|$.

We can obtain a different time complexity by using the parametric minimum cut algorithm (Gallo et al. [11]). The parametric minimization problem

$$\text{minimize } \gamma - \alpha b \ \text{ for all } \ \alpha \geq 0$$

corresponds to the parametric minimum cut problem

$$\text{find minimum } s\text{-}t \text{ cuts in } \mathcal{G}^-_\alpha \ \text{ for all } \ \alpha \geq 0.$$

To solve this parametric cut problem, we can utilize the parametric minimum cut algorithm [11] (see also [16]). We remark that the directed graph $\mathcal{G}^-_\alpha$ satisfies the monotonicity in $\alpha \geq 0$ in the meaning of [11]. As a result, we have the following time complexity.

Theorem 7. *For a generalized graph cut function* $\gamma : 2^V \to \mathbb{R}$ *generated from* $\mathcal{G} = (\{s\} \cup \{t\} \cup V \cup U, \mathcal{E})$, *problem* (9) *can be solved in* $\mathrm{O}((n+|U|)|\mathcal{E}| \log \frac{(n+|U|)^2}{|\mathcal{E}|})$ *time, where* $n = |V|$.

The algorithm of Theorem 7, which is much faster than that of Theorem 6 from a theoretical point of view, is rather complicated to implement.

In view of Theorem 1, we can solve the convex minimization problem under constraints with respect to the structured submodular function γ,

$$\min_{\boldsymbol{x} \in \mathrm{B}(\gamma)} \sum_{i \in V} w_i(x_i)$$

in $O(n(n+|U|)|\mathcal{E}|)$ or $O((n+|U|)|\mathcal{E}|\log\frac{(n+|U|)^2}{|\mathcal{E}|})$ time for a number of separable convex objective functions.

7 Experimental results

We investigated the empirical performance of the proposed scheme using synthetic and real-world datasets. In Section 7.1, we compare the proposed method in the application to proximal methods for structured regularized least-squares regression, with the state-of-the-art algorithms. In Section 7.2, we apply the proposed algorithm to the densest subgraph problem for large real web-network data. The experiments below were run on a 2.3 GHz 64-bit workstation using Matlab with Mex implementations. And we used SPAMS (SPArse Modeling Software) [18] for the implementations of the proximal methods for the first experiment.

7.1 Comparison in proximal methods

In the first experiment, we compared the proposed algorithm in the application to proximal methods with the state-of-the-art algorithms. As for the regularization term, we used fused-regularization $\Omega_{\text{fused}}(\boldsymbol{\beta})$ and group regularization (with l_∞-norm) $\Omega_{\text{group}}(\boldsymbol{\beta})$ (for a given set of groups $\mathcal{G}$), respectively represented as

$$\Omega_{\text{fused}}(\boldsymbol{\beta}) = \textstyle\sum_{i=1}^{n-1}|\beta_i - \beta_{i+1}| \text{ and}$$
$$\Omega_{\text{group}}(\boldsymbol{\beta}) = \textstyle\sum_{g\in\mathcal{G}} d_g\|\boldsymbol{\beta}_g\|_\infty,$$

where d_g is the weight of the group g. As the comparison partners, we used the proximal methods for the above regularization; the one based on the homotopy algorithm for $\Omega_{\text{fused}}(\boldsymbol{\beta})$ [7] (Homo.) and the one by Mairal et al. [19] for $\Omega_{\text{group}}(\boldsymbol{\beta})$ (NFA), as well as the minimum-norm-point algorithm (MNP) for the calculation of the proximal operator. Since both regularizations can be represented as the decomposable submodular function, we applied the parametric flow algorithm for computing the proximal operators (DA).

We generated data as follows. First for the evaluation with fused regularization, one feature is first selected randomly and the next one is selected with probability 0.4 from each neighboring feature or with probability $0.2/(N-2)$ from the remaining ones and repeat this procedure until k features are selected. For group regularization, the features are covered by 20–200 overlapping groups of size 15. The causal features are chosen to be the union of 2 of these groups. Here, we assign weights $d_g = 2$ to those causal groups and $d_g = 1$ to all other groups. We then simulate N data points $(\mathbf{x}(i), y(i))$, with $y(i) = \bar{\boldsymbol{\beta}}^\top\mathbf{x}(i) + \epsilon$, $\epsilon \sim \mathcal{N}(0,\sigma^2)$, where $\bar{\boldsymbol{\beta}}$ is 0 for non-causal features and normally distributed otherwise.

Table 1: Comparison of running time (seconds) for the proposed and existing methods.

n	N	k	DA	MNP	Homo.
500	500	20	0.024	5.083	0.084
500	1,000	20	0.062	146.969	0.531
500	5,000	20	1.085	—	32.676
1,000	500	20	0.019	3.891	0.058
1,000	1,000	20	0.059	98.310	0.266
1,000	5,000	20	1.064	—	12.372

n	N	k	DA	MNP	NFA
500	500	$\sim$20	0.021	8.910	0.015
500	1,000	$\sim$20	0.056	280.117	0.052
500	5,000	$\sim$20	1.091	—	1.112
1,000	500	$\sim$20	0.020	6.108	0.015
1,000	1,000	$\sim$20	0.054	198.010	0.051
1,000	5,000	$\sim$20	1.003	—	0.896

Since all methods calculate the same objectives in principle, here we report only the comparison of the empirical running time. Tables 1 show the running time by the algorithms for reaching the duality-gap within 10^{-4}, averaged over 20 datasets each. We can see that the algorithms based on the parametric-flow algorithm, including ours, run much faster than the others. Note that our scheme can be applied to more general form of structured regularization (Eq. (5) for the graph cut implementation) than $\Omega_{\text{fused}}(\boldsymbol{\beta})$ and $\Omega_{\text{group}}(\boldsymbol{\beta})$.

7.2 Densest subgraphs in web graphs

In the second experiment, we applied the proposed algorithm to the densest subgraph problem using public web-graph and social-network datasets [5]. The characteristics of each data set are shown in Table 2. Although the minimum-norm-point algorithm was applied to the same problem on one of the datasets (*cnr-2000*) in [23], the data was sub-sampled to 5,000 nodes due to its computational cost. However, in this experiment, we used the full datasets for the analyses, which was possible because our framework runs much more efficiently than the algorithm in [23].

The running time for applying our method to each dataset is shown in Table 2 as well as the number of optimal solutions found by the algorithm. Our method could find exactly optimal-solutions for several k for these large datasets in practical time. Note again that, if k is fixed beforehand, the densest subgraph problem with the size constraint is NP-hard and thus there is no efficient algorithm. Also, the graphs in Figure 6 show plot examples of intensity $I(\mathcal{S})$ versus the sizes of subsets k found by the algorithm. The tendency seems to be that our methods can find more optimal solutions if graphs are denser.

Table 2: Resulting running-time and the number of optimal subsets found by the algorithm as well as the characteristics of datasets.

Data	# Node	# Arc	Time [s]	# Set
cnr-2000	325,557	3,216,152	20.55	22
uk-2007	100,000	3,050,615	19.70	49
in-2004	1,382,908	16,917,053	225.90	5,971
eu-2500	862,664	19,235,140	278.50	4,933
wordassoc.	10,617	72,172	0.15	2
amazon-2008	735,323	5,158,388	127.51	1,882
dblp-2010	326,186	1,615,400	19.68	985
dblp-2011	986,324	6,707,236	96.60	979

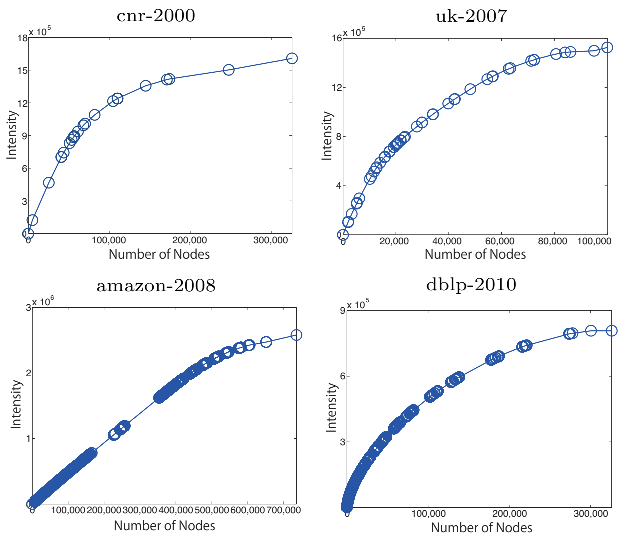


Figure 6: Example plots of $I(\mathcal{S})$ versus k for *cnr-2000*, *uk-2007*, *amazon-2008* and *dblp-2010*.

8 Concluding remarks

We have shown that when a submodular function f has a directed graph representation the separable convex minimization problem under submodular constraints can be solved pretty efficiently compared to general submodular optimization methods. It is known that quite a lot of submodular functions have graph structures (refer to Jegelka, Lin, and Bilmes [14]). The proposed methods are based on the general theory of submodular functions and (parametric) maximum flow algorithms. In addition, we remark that the proposed methods can deal with various essentially equivalent objective functions for the problem.

Appendix: Submodularity of generalized graph cut functions

In order to make this paper self-contained, we give a proof of the submodularity of a generalized graph cut function [14], $\gamma : 2^V \to \mathbb{R}$ defined in (4).

We set $\beta \geq 0$ as the sum of all edge capacities of $\mathcal{G}$. Let $\mathbf{1} \in \mathbb{R}^n$ be the all-one vector and let $1 : 2^V \to \mathbb{R}$ be a set function defined by $1(S) = |S|$ for each $S \subseteq V$. The directed graph $\mathcal{G}^+_{\beta\mathbf{1}}$ (see Section 6) generates the set function $\gamma + \beta 1$. For each $S \subseteq V$, let $\gamma'(S)$ be the minimum capacity of a cut separating $\{s\} \cup S$ from the sink t in $\mathcal{G}^+_{\beta\mathbf{1}}$. By the result of Megiddo [20] on network flow problems with multiple terminals, the set function $\gamma' : 2^V \to \mathbb{R}$ is submodular. For each $S \subseteq V$, we have

$$\begin{aligned}
\gamma'(S) &= \min_{W \subseteq (U \cup V \setminus S)} \sum \{c(e) : e \in \delta^{\text{out}}_{\mathcal{G}^+_{\beta\mathbf{1}}}(\{s\} \cup S \cup W)\} \\
&= \min_{W \subseteq (U \cup V \setminus S)} \Big(\sum \{c(e) : e \in \delta^{\text{out}}_{\mathcal{G}}(\{s\} \cup S \cup W)\} \\
&\qquad\qquad + \beta|S| + \beta|W \cap (V \setminus S)| \Big) \\
&= \beta|S| + \min_{W \subseteq U} \sum \{c(e) : e \in \delta^{\text{out}}_{\mathcal{G}}(\{s\} \cup S \cup W)\} \\
&= \beta|S| + \gamma(S),
\end{aligned}$$

where the third equality holds because β is sufficiently large. Since $\gamma' = \gamma + \beta 1$ is submodular, the function γ is also submodular.

Acknowledgments

This research was partially supported by Aihara Project, the FIRST program from JSPS, JST ERATO Kawarabayashi Large Graph Project, JST PRESTO PROGRAM (Synthesis of Knowledge for Information Oriented Society), and the Cooperative Research Program of "Network Joint Research Center for Materials and Devices".

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, 1993.

[2] F. Bach. Structured sparsity-inducing norms through submodular functions. In *Advances in Neural Information Processing Systems 23 (NIPS 2010)*, pages 118–126, 2010.

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[4] A. Chambolle and J. Darbon. On total variation minimization and surface evolution using parametric maximal flows. *International Journal of Computer Vision*, 84(3), 2009.

[5] P. Boldi et al. Laboratory for Web Algorithmics. `http://law.di.unimi.it/datasets.php`.

[6] L. Fleischer and S. Iwata. A push-relabel framework for submodular function minimization and applications to parametric optimization. *Discrete Appl. Math.*, 131:311–322, 2003.

[7] J. Friedman, T. Hastie, H. Holfling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of statistics*, 1(2):302–332, 2007.

[8] S. Fujishige. Lexicographically optimal base of a polymatroid with respect to a weight vector. *Mathematics of Operations Research*, 5:186–196, 1980.

[9] S. Fujishige. *Submodular Functions and Optimization*. Elsevier, 2nd edition, 2005.

[10] S. Fujishige, T. Hayashi, and S. Isotani. The minimum-norm-point algorithm applied to submodular function minimization and linear programming. Technical report, Research Institute for Mathematical Sciences Preprint RIMS-1571, Kyoto University, Kyoto, Japan, 2006.

[11] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18:30–55, 1989.

[12] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM*, 35:921–940, 1988.

[13] S. Iwata and K. Nagano. Submodular function minimization under covering constraints. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2009)*, pages 671–680, 2009.

[14] S. Jegelka, H. Lin, and J. Bilmes. On fast approximate submodular minimization. In *Advances in Neural Information Processing Systems 24 (NIPS 2011)*, pages 460–468, 2011.

[15] Pushmeet Kohli, Lubor Ladicky, and Philip H.S. Torr. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision*, 82:302–324, 2009.

[16] V. Kolmogorov. A faster algorithm for computing the principal sequence of partitions ofa graph. *Algorithmica*, 56:394–412, 2010.

[17] L. Lovász. Submodular functions and convexity. In A. Bachem, M. Grötschel, and B. Korte, editors, *Mathematical Programming — The State of the Art*, pages 235–257. Springer-Verlag, 1983.

[18] J. Mairal, F. Bach, J. Ponce, G. Sapiro, R. Jenatton, and G. Obozinski. SPArse Modeling Software. `http://spams-devel.gforge.inria.fr/`.

[19] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Convex and network flow optimization for structured sparsity. *Journal of Machine Learning Research*, 12:2681–2720, 2011.

[20] N. Megiddo. Optimal flows in networks with multiple sources and sinks. *Mathematical Programming*, 7:97–107, 1974.

[21] K. Nagano. A faster parametric submodular function minimization algorithm and applications. Technical report, METR 2007-43, University of Tokyo, Tokyo, Japan, 2007.

[22] K. Nagano and K. Aihara. Equivalence of convex minimization problems over base polytopes. *Japan journal of industrial and applied mathematics*, 29:519–534, 2012.

[23] K. Nagano, Y. Kawahara, and K. Aihara. Size-constrained submodular minimization through minimum norm base. In *Proceedings of the 28th International Conference on Machine Learning (ICML 2011)*, pages 977–984, 2011.

[24] K. Nagano, Y. Kawahara, and S. Iwata. Minimum average cost clustering. In *Advances in Neural Information Processing Systems 23 (NIPS 2010)*, pages 1759–1767, 2010.

[25] M. Narasimhan, N. Jojic, and J. Bilmes. Q-clustering. In *Advances in Neural Information Processing Systems 18 (NIPS 2005)*, pages 979–986, 2005.

[26] Yu. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103:127–152, 2005.

[27] J. B. Orlin. A faster strongly polynomial time algorithm for submodular function minimization. *Mathematical Programming*, 118:237–251, 2009.

[28] J. B. Orlin. Max flows in O(nm) time or less. In *Proceedings of the 45th ACM Symposium on the Theory of Computing (STOC 2013)*, 2013. To appear.

[29] M. Queyranne. Minimizing symmetric submodular functions. *Mathematical Programming*, 82:3–12, 1998.

[30] A. Schrijver. *Combinatorial Optimization — Polyhedra and Efficiency*. Springer-Verlag, 2003.

[31] P. Stobbe and A. Krause. Efficient minimization of decomposable submodular functions. In *Advances in Neural Information Processing Systems 23 (NIPS 2010)*, pages 2208–2216, 2010.

Treedy: A Heuristic for Counting and Sampling Subsets

Teppo Niinimäki
Helsinki Institute for Information Technology
Deparment of Computer Science
University of Helsinki
teppo.niinimaki@cs.helsinki.fi

Mikko Koivisto
Helsinki Institute for Information Technology
Deparment of Computer Science
University of Helsinki
mikko.koivisto@cs.helsinki.fi

Abstract

Consider a collection of weighted subsets of a ground set N. Given a query subset Q of N, how fast can one (1) find the weighted sum over all subsets of Q, and (2) sample a subset of Q proportionally to the weights? We present a tree-based greedy heuristic, Treedy, that for a given positive tolerance d answers such counting and sampling queries to within a guaranteed relative error d and total variation distance d, respectively. Experimental results on artificial instances and in application to Bayesian structure discovery in Bayesian networks show that approximations yield dramatic savings in running time compared to exact computation, and that Treedy typically outperforms a previously proposed sorting-based heuristic.

1 INTRODUCTION

Reasoning with probabilistic models typically deals with queries on sets of weighted points. For instance, one may ask a maximum-weight point subject to the constraint that the point satisfies some given property. Likewise, one may ask the total weight (e.g., the probability mass) of the points or a random point sampled proportionally to the weights, subject to the given constraint. The large number of points and the complexity of the constraint often render these tasks computationally very challenging.

The present works addresses a particular class of queries, we call *subset queries*, formalized as follows. Let N be a ground set of n elements and $\mathcal{C}$ a collection of m subsets of N, each subset $S \in \mathcal{C}$ associated with a non-negative weight $w(S)$. For convenience, we extend the weight function to all subsets of the ground set by letting $w(S) = 0$ for $S \notin \mathcal{C}$. A *subset counting query* asks the total weight of all subsets of a given query subset $Q \subseteq N$, given by

$$W(Q) = \sum_{S \subseteq Q} w(S)\,. \qquad (1)$$

Analogously, a *subset sampling query* asks a random subset of the query subset Q, such that any particular subset $S \subseteq Q$ gets selected with probability $w(S)/W(Q)$. While queries about maxima, minima, median, etc. can be defined in a similar fashion, we will restrict ourselves to counting and sampling queries in the sequel.

Our focus will be on scenarios where m, the size of the collection, is much smaller than 2^n, the number of all subsets of the ground set. The opposite case when m is close to 2^n is also fundamental but, to a large extent, well understood. Namely, using the fast zeta transform algorithm (Yates, 1937; Kennes and Smets, 1990; Kennes, 1991; Koivisto and Sood, 2004; Koivisto, 2006) the values $W(Q)$ can be computed for all $Q \subseteq N$ in a total of roughly $n2^n$ additions, after which answering any counting or sampling query is very fast.

1.1 APPLICATION: ORDER-MCMC FOR BAYESIAN NETWORK LEARNING

A motivating example of subset queries is provided by the order-MCMC method of Friedman and Koller (2003) for learning the directed acyclic graph (DAG) of a Bayesian network model (Pearl, 1988, 2000; Buntine, 1991; Heckerman et al., 1995). For closely related recent developments that likewise involve subset queries, see the works of Ellis and Wong (2008), Niinimäki et al. (2011), and Niinimäki and Koivisto (2013).

Order-MCMC samples node orderings by simulating a Markov chain whose stationary distribution is the posterior distribution. The time consumption of order-MCMC is determined by the complexity of evaluating the posterior probability (up to a normalizing constant) of a given node ordering $v_1 v_2 \cdots v_n$. Efficient evaluation is facilitated by the factorization of the posterior into a product $W_1 W_2 \cdots W_n$, where each factor W_j is given by (1) as the total weight $W(\{v_1, v_2, \ldots, v_{j-1}\})$ for a weight function w that depends on the node v_j. The interpretation is that any subset of the nodes preceding v_j can form the set of *parents* of v_j in the network, the weight indicating how well a particular selection of parents fits the data and the prior beliefs.

To quantify the time requirements in this example, suppose that each node is allowed to have at most k parents—having a relatively small k is a common practice unless n is very small. Then each W_j can be evaluated using about $\binom{j-1}{k}_* = \binom{j-1}{0} + \binom{j-1}{1} + \cdots + \binom{j-1}{k}$ additions, and the posterior using about $\binom{0}{k}_* + \binom{1}{k}_* + \cdots + \binom{n-1}{k}_* = \binom{n}{k+1}_* - 1 > \binom{n}{k+1}$ additions (followed by $n-1$ multiplications). This is tolerable for small values of n and k, but for larger values, say $n = 60$ and $k = 5$, the time requirement becomes infeasible in practice, for the computations are performed for thousands of node orderings.

The same concern holds for the possible second phase of the order-MCMC method, in which each sampled node ordering is used for sampling some number of DAGs compatible with the ordering. This amounts to n subset sampling queries per DAG.

1.2 AN APPROXIMATION APPROACH

One might hope for a data structure that enables rapid answering of subset queries. Trivially, counting queries can be answered in $O(n)$ time by precomputing and storing the answers to all possible 2^n queries in advance. But this approach becomes soon unfeasible for larger n, especially due to the large memory requirement. It is to be contrasted with the other extreme approach: visit all members $S \in \mathcal{C}$ and add $w(S)$ to the sum if $S \subseteq Q$, taking $O(mn)$ time and essentially no extra memory. Whether there are efficient ways to trade memory for time, is an open question. However, there is strong negative evidence associated with closely related existence queries ("Does the query set contain some set in $\mathcal{C}$?"). Namely, the best known tradeoffs are rather inefficient and of theoretical interest only (Charikar et al., 2002), and lower bounds that suggest the impossibility of finding much better tradeoffs are known (Pătraşcu, 2011).

Given the state of affairs concerning exact solutions, we in this work settle for approximations to reduce the time requirement of subset queries. In counting queries we allow an additive relative error d. Likewise, in sampling queries we require the sampling distribution be at a total variation distance at most d from the exact distribution. Here d is a parameter that can be set to close to zero, say $d = 0.01$, to guarantee very accurate approximations. In the order-MCMC application, for instance, it is easy to verify that accurate approximations to subset queries translate, in a straightforward manner, to accurate approximations at the end results of posterior inference.

The idea of approximation is not new. Indeed, the starting point of the present study is the following heuristic by Friedman and Koller (2003) to speed up order-MCMC: Given a query set Q, visit a fixed number m' of heaviest members $S \in \mathcal{C}$ in decreasing order by weight, and if $S \subseteq Q$, then increase the sum by the weight of S. Finally, return the accumulated sum, unless the largest counted weight fails to be some factor γ larger than the smallest (last) weight in the list, in which case compute the sum by brute-force enumeration of all subsets of Q in $\mathcal{C}$. The rationale is that for large Q it is likely that there is at least one heavy $S \subseteq Q$ among the m' heaviest sets in $\mathcal{C}$, and thus the brute-force phase is avoided. Choosing a large enough γ guarantees that the lost mass is negligible. Whether this heuristic is close to the best possible, has remained an open question.

Here, we address the question in several ways. Our focus is exclusively on *collector* algorithms that, like the aforementioned heuristic by Friedman and Koller, are based on visiting some of the subsets of the ground set in some order, adding up the weights of those that are subsets of the query set, and stopping using some appropriate rule. We begin in Section 2 by showing how any algorithm of this type for approximative counting also yields a sampling algorithm with a corresponding accuracy. The section continues by describing two extreme approaches to counting queries: a brute-force algorithm *Exact* that produces the exact value, and an idealized algorithm *Ideal* that only visits the minimum number of heaviest subsets that suffice for the desired approximation error, so providing us a lower bound for the amount of work needed by any collector algorithm. We also streamline the heuristic of Friedman and Koller by formulating a stronger stopping rule that achieves the same accuracy guarantees with less work. We call the resulting algorithm *Sorted*.

Our main contribution is a novel heuristic, presented in Section 3. The motivation of the heuristic stems from the observation that *Sorted* becomes slow when the heaviest subsets are not contained by the query set. Our idea is to restrict the search to subsets of the query set, however, turning the brute-force enumeration into a controlled approximation algorithm that, in a greedy fashion, aims to visit first subsets that are "likely" to be heavier. As the enumeration proceeds from smaller subsets to larger ones in a tree-structured manner, we call the algorithm *Treedy*.

We compare the heuristics to the exact and the idealized algorithm in Section 4. We report on experiments with synthetic instances and in application to Bayesian network learning using order-MCMC.

To keep the presentation simple and succinct, we will assume that the collection $\mathcal{C}$ is *downward closed.* That is, we assume that $\mathcal{C}$ equals the *downward closure* $\mathcal{C}_* = \{T : T \subseteq S \text{ for some } S \in \mathcal{C}\}$. In the experiments we further restrict our attention to the case where $\mathcal{C}$ consists of all $S \subseteq N$ with $|S| \leq k$ for some relatively small k. These restrictions are, however, not crucial for the validity of the studied methods. We discuss this issue, among other things, in Section 5.

2 PRELIMINARIES

Throughout this section and the remainder of the paper, we consider a weighted downward closed collection $\mathcal{C}$ of m subsets of some n-element ground set N. For a query set $Q \subseteq N$, we call a set S *relevant* if $S \in \mathcal{C}$ and $S \subseteq Q$. We denote the collection of relevant sets by $\mathcal{C}_Q$. If Q is clear from the context, we may denote the total weight $W(Q)$ simply by W. We will denote by d the approximation *tolerance*, $0 \leq d \leq 1$, whether referring to an upper bound for the additive relative error or for the total variation distance.

2.1 FROM COUNTING TO SAMPLING

Below is a generic algorithm that uses a collector algorithm for counting queries (the first step) to answer sampling queries (the second step).

Algorithm *Draw*
Given a query set Q and tolerance d, do the following:

D1 Visit some relevant sets $S_1, S_2, \ldots, S_r$ whose total weight W' is at least $(1-d)W$. Store the cumulative sums $W_i = \sum_{j=1}^{i} w(S_j)$.

D2 Draw a random variable U from the uniform distribution on the interval $[0, W']$. Find an i such that $W_{i-1} < U \leq W_i$ and return S_i.

It is easy to see that *Draw* returns a set S with probability $\pi'(S)$ that satisfies $\pi'(S) = w(S)/W'$ if S is a visited relevant set and $\pi'(S) = 0$ otherwise. The next result shows that this guarantees a small deviation from the exact distribution $\pi(S) = w(S)/W$, as measured by the commonly used total variation distance. The *total variation distance* between two probability measures μ and μ' on a finite set Ω is defined as $\delta(\mu, \mu') = \max_{A \subseteq \Omega} |\mu(A) - \mu'(A)|$ and can be simplified to $\delta(\mu, \mu') = \sum_{a \in \Omega} |\mu(a) - \mu'(a)|/2$. We attribute the following theorem to folklore; the proof is elementary and included here for convenience.

Theorem 1. *The total variation distance between the above defined π and π' is at most d.*

Proof. As $\pi'(S) \geq \pi(S)$ for each visited relevant set S and $\pi'(S) = 0$ otherwise, we have

$$\begin{aligned} 2\delta(\pi, \pi') &= \sum_{i=1}^{r} \left[\pi'(S_i) - \pi(S_i)\right] + 1 - \sum_{i=1}^{r} \pi(S_i) \\ &= 2 - 2W'/W\,. \end{aligned}$$

Using $W' \geq (1-d)W$ completes the proof. □

If the cumulative weights W_i are stored in a simple array indexed by i, step D2 can be implemented to run in $O(\log r)$ time by using binary search. However, if the distribution π' has small entropy H (i.e., the mass is concentrated on some subsets), then much faster implementation running in roughly $O(H)$ time is possible by using, e.g., a Huffman coding based data structure.

The running time of *Draw* is clearly dominated by step D1. This, in part, motivates the investigations of efficient collector algorithms for counting queries.

2.2 EXACT AND IDEAL COUNTING

There are two obvious brute-force approaches to compute the exact total weight of the query set Q. One is to visit every set S in the given collection $\mathcal{C}$ and add the weight of S to a cumulative sum if $S \subseteq Q$. The other approach is to only visit sets $S \subseteq Q$ and add the weight of S to a cumulative sum if $S \in \mathcal{C}$. The following algorithm assumes the latter approach:

Algorithm *Exact*
Given a query set Q, do the following:

E1 Visit the relevant sets in lexicographic order and return the sum of their weights.

For downward closed collections $\mathcal{C}$ it is, in fact, more efficient to implement the algorithm so that the membership test $S \in \mathcal{C}$ is avoided, at the cost of visiting also a few sets that are not subsets of Q. The idea is to use a data structure where each member S of $\mathcal{C}$ is linked to its one-element larger successors $S \cup \{x\} \in \mathcal{C}$, with the link labeled by the element x. Namely, then the algorithm can proceed in the lexicographic order at the cost of testing whether $x \in Q$ also for some some irrelevant sets $S \cup \{x\} \not\subseteq Q$. Technically, this makes the number of visited sets generally exceed the number of relevant sets $|\mathcal{C}_Q|$; however, the extra visits are very quick due to the simplicity of the test.

We note that when $\mathcal{C}$ consists of all subsets of size at most some k, then it is easy to visit only the relevant sets and avoid the aforementioned technicalities.

When an approximation of the total weight suffices, the performance of *Exact* is no longer close to the best possible. An ideal collector algorithm would visit as few as possible sets necessary for gathering the required proportion of the mass:

Algorithm *Ideal*
Given a query set Q and tolerance d, do the following:

I1 Visit, in some order, the minimum number of the heaviest relevant sets whose total weight W' is at least $(1-d)W$. Return W'.

We should note that *Ideal* is an "idealized" algorithm in the sense that we do not know how to efficiently find the minimum number of the heaviest relevant sets. Nevertheless, we can estimate the running time of the algorithm under the supposition that a list of sufficient relevant sets is available to the algorithm for free.

2.3 COUNTING BY PRE-SORTING

If the query set Q is large, then one can achieve almost ideal performance by visiting sufficiently many members of $\mathcal{C}$ in decreasing order by weight. Sorting needs to be done only in the initialization phase, before any query. For smaller Q the heaviest sets of $\mathcal{C}$ are, however, likely to include also irrelevant sets that are not subsets of Q. Then the number of visited sets may grow much larger than $|\mathcal{C}_Q|$. Therefore it is advisable to switch over to the exact algorithm after about $|\mathcal{C}_Q|$ visited sets. To keep the number of visited sets as small as possible, it is also crucial to devise an efficient stopping rule. The following algorithm adapts these ideas and implements a stopping condition that is stronger than in the original formulation by Friedman and Koller (2003):

Algorithm *Sorted*

S0 Before any query, sort the sets in $\mathcal{C}$ into decreasing order by weight, $w(S_1) \geq w(S_2) \geq \cdots \geq w(S_m)$. Store the cumulative sums $W_i = \sum_{j=i+1}^{m} w(S_j)$.

S1 Given a query set Q and tolerance d, initialize a counter for yet nonvisited relevant sets $t = |\mathcal{C}_Q|$, a step counter $j = 0$, and $W' = 0$, and do the following: increase j by 1; if $j \geq |\mathcal{C}_Q|$, then switch to *Exact*; if $S_j \subseteq Q$, then add $w(S_j)$ to W' and decrease t by 1; if $W' \geq (1-d)(W' + W_j - W_{j+t})$, then stop and return W'.

Theorem 2. *Algorithm Sorted is correct.*

Proof. If the algorithm switches to *Exact* at some point, then the correctness of the algorithm is clear. Suppose therefore that the algorithm stops when the condition $W' \geq (1-d)(W' + W_j - W_{j+t})$ is satisfied. It suffices to show that $W' + W_j - W_{j+t} \geq W$. To this end, observe that W' is a sum of the weights of the $|\mathcal{C}_Q| - t$ heaviest relevant sets, and that $W_j - W_{j+t}$ is at least as large as the sum of the t remaining (lightest) relevant sets. □

Table 1: Processing a Subset Counting Query. For each algorithm the visiting order of subsets is shown, for ground set {A, B, C, D}, query set {B, C, D}, and approximation tolerance 20 %.

$S \in \mathcal{C}$	$w(S)$	*Exact*	*Ideal*	*Sorted*	*Treedy*
AB	99			1	
AD	90			2	
A	85	2		3	2
∅	80	1	✓	4	1
B	70	3	✓	5	3
AC	60			6	
D	50	8	✓	7	4
BD	14	5			5
C	13	6			6
CD	12	7			
BC	11	4			

See Table 1 for an illustration and comparison to *Exact* and *Ideal*. Note that in the example, *Sorted* achieves the desired approximation guarantee after visiting 7 sets; it does not switch to *Exact*.

3 THE TREEDY HEURISTIC

We next present a novel heuristic, *Treedy*, for approximate counting queries. Like *Exact*, the heuristic operates on a lexicographically structured tree. The *lexicographical tree* of $\mathcal{C}$ is a rooted tree on the collection $\mathcal{C}$, defined as follows. The empty set is the root of the tree. Any other set S is a *son* of another set S' if $S = S' \cup \{x\}$ where x is the last element in S in alphabetical order; the notions of a *brother*, *ancestor*, and *descendant* are defined in the obvious way. For each set $S \in \mathcal{C}$ define the *weight potential* $\phi(S)$ as the sum of the weights of all descendants of S (including S itself). In the initialization phase, *Treedy* modifies this structure appropriately, and in the query phase it proceeds in a greedy fashion.

Algorithm *Treedy*

T0 Before any query, modify the lexicographical tree of $\mathcal{C}$ into a *greedy tree* as follows: For each set, sort its sons in decreasing order by their weight potentials. Then remove the links to all but the

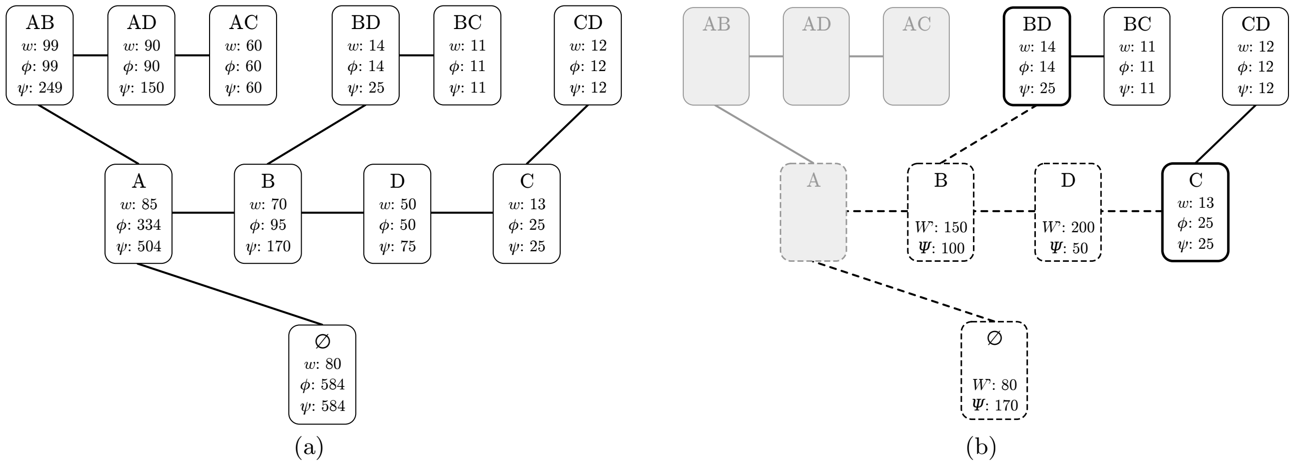


Figure 1: *Treedy* in action on the example instance described in Table 1. (a) The greedy tree built in step T0 is shown. (b) The situation at the end of step T1 is shown. Dashed lines connect the visited sets. The sets marked by thicker rectangles constitute the collection $\mathcal{R}$ at the end of step T1. The last set removed from $\mathcal{R}$ is {D}. A subtree that is discarded once found to contain only irrelevant sets is shown in gray. Note that ϕ and ψ remain unchanged after step T0; only the values of W' and $\Psi(\mathcal{R})$ change during the execution of step T1 (for each set shown are the values just after removing the set from $\mathcal{R}$ and updating W').

first son and create a linked list to connect the brothers in this order. As a result, each set has (at most) two links: one to its brother with next largest weight potential and the other to its son with the largest weight potential. Finally compute the *aggregate potential* $\psi(S)$ for each set S by summing the weight potentials of S and its subsequent smaller brothers.

T1 Given a query set Q and tolerance d, initialize $W' = 0$ and $\mathcal{R} = \{\emptyset\}$ and repeat the following: remove from $\mathcal{R}$ a set S with the largest aggregate potential; add $w(S)$ to W'; add the next relevant brother (if any) of S into $\mathcal{R}$ and ignore the preceding irrelevant brothers (and their descendants); add the first relevant son (if any) of S into $\mathcal{R}$ and ignore the preceding irrelevant sons (and their descendants); if $W' \geq (1-d)\big(W' + \Psi(\mathcal{R})\big)$, where $\Psi(\mathcal{R}) = \sum_{S\in\mathcal{R}} \psi(S)$ is the aggregate potential of $\mathcal{R}$, then stop and return W'.

See Table 1 and Figure 1 for an illustration of an execution of *Treedy*. In that small example, the savings of *Treedy* compared to *Exact* are rather modest. We leave it to the reader to imagine how larger savings are possible on larger problem instances.

Theorem 3. *Algorithm Treedy is correct.*

Proof. First note that, during the execution of the algorithm, $\mathcal{R}$ contains only relevant sets and thus only weights of relevant sets are added to W'. Therefore, it is sufficient to show that the invariant $W' + \Psi(\mathcal{R}) \geq W(Q)$ holds during the execution of the algorithm; correctness then follows from the stopping condition.

To see that the invariant holds, observe that in the beginning $\Psi(\mathcal{R})$ is the sum of the weights of all relevant sets plus the weights of all irrelevant sets. In each step the algorithm removes a set S from $\mathcal{R}$ but adds back the next relevant brother of S and the first relevant son of S. The aggregate potential of $\mathcal{R}$ is thus decreased by $w(S)$ and by the weight potentials of any possibly ignored irrelevant brothers and sons. Since the descendants of irrelevant sets are also irrelevant, the only relevant set whose weight is subtracted from $\Psi(\mathcal{R})$ is S. As $w(S)$ is added to W', no weight of a relevant set is removed from $W' + \Psi(\mathcal{R})$. Thus, by the induction principle the invariant holds during the execution of the algorithm. □

The potential efficiency of *Treedy* stems from the fact that, contrary to *Sorted*, none of the descendants of an irrelevant set will be visited. On the other hand, the algorithm is allowed to visit some irrelevant sets to enable efficient implementation of the greedy best-first order. But visiting irrelevant nodes is fast since those are just ignored; no update of $\mathcal{R}$, $\Psi(\mathcal{R})$ or W' is needed.

The greedy tree can be constructed in a straightforward manner in $O(m \log n)$ time, for sorting the m_S sons of each set $S \in \mathcal{C}$ takes $O(m_S \log m_S)$ time, $m_S \leq n$, and $\sum_{S\in\mathcal{C}} m_S = m - 1$. Thus the initialization cost is essentially linear in the input size and negligible when the number of counting queries is large.

To implement the query algorithm efficiently, we have to overcome two challenges: (1) how to store $\mathcal{R}$ and (2) how to avoid computing $\Psi(\mathcal{R})$ from scratch in every step. We address the first challenge by keeping the members of $\mathcal{R}$ in a binary heap, which enables updating $\mathcal{R}$ in $O(\log |\mathcal{R}|)$ time per step of the algorithm.

To address the second challenge, we maintain the sum $\Psi(\mathcal{R})$ during the execution of the algorithm by, in each step, subtracting from $\Psi(\mathcal{R})$ the aggregate potential of the set removed from $\mathcal{R}$ and adding to $\Psi(\mathcal{R})$ the aggregate potentials of the sets added to $\mathcal{R}$. However, if the relative differences of the weights $w(S)$ are large, then this approach can lead to problems with numerical accuracy. If this is the case, a solution is to not maintain $\Psi(\mathcal{R})$ at all until we know that the stopping condition is relatively close to hold. More specifically, suppose $\Psi(\mathcal{R})$ is computed from scratch and maintained only after the aggregate potential of the set removed from $\mathcal{R}$ gets smaller than dW'. Then we know that $\Psi(\mathcal{R})$ has to decrease at most by the factor $|\mathcal{R}|$ before the stopping condition is met, and thus the accuracy problems should be gone.

4 EXPERIMENTAL STUDIES

We have implemented the presented algorithms in the C++ language.[1] We next report on experimental studies on artificially generated instances of subset query problems as well as several instances of the Bayesian network learning application using our implementation of the order-MCMC method of Friedman and Koller (2003). The focus of the experimental studies is in investigating the relationship of running time and approximation guarantee of the four algorithms.

4.1 ARTIFICIAL INSTANCES

We generated various weight functions $w(S)$ on subsets $S \subseteq N$ of size at most k. We varied $n = |N|$ in $\{20, 60\}$ and k in $\{3, 5\}$. We considered four types of weight functions, each based on the following building block:

$$w(S) = \exp\left(\lambda \sum_{i \in S} U_i\right), \quad U_i \overset{\text{iid}}{\sim} \text{Uniform}(\kappa - 1, \kappa)\,.$$

Here λ is a parameter that specifies the variance of the weights; the larger the λ, the larger the variance. The parameter κ specifies the (expected) number of elements of the ground set that contribute positively to the weight. Note that the weight of the empty set is always 1. The four types are the following:

Flat: $\lambda = 10$, $\kappa = k/n$.

Steep: $\lambda = 200$, $\kappa = k/n$.

Mixture: Take the sum of 5 flat and 5 steep ones, for both types letting the product κn take the 5 values $k-1$, k, $k+1$, $k+2$, and $k+3$. This creates 10 distinct "local maxima".

Shuffled: Like in the mixture type but permuting the weights $w(S)$ randomly among the subsets of same size. This destroys the dependence of the weights of subsets that have large intersection.

For each of the four types and the values of n and k, we generated 5 random weight functions, executed the algorithms *Exact*, *Ideal*, *Sorted*, and *Treedy*, for 1000 query sets sampled uniformly at random for each query set size from 1 to n. Figure 2 shows average running times.

We see that for flat weight functions, the approximation algorithms yield significant speedups over the exact algorithm only when the approximation tolerance is relatively large. For steep weight functions, as well as for mixtures and shuffled mixtures, the speedups are however by two orders or magnitude already with small approximation tolerance. The speedups become the more dramatic, the larger the n and k are. Examining the effect of the query set size reveals that *Sorted* performs better than *Treedy* for larger query sets, but for smaller query sets *Treedy* is faster. For larger values of n and k, *Treedy* outperforms *Sorted* by a factor of about 5. To our surprise, shuffling the weight function has essentially no effect to the performance of *Treedy*.

4.2 APPLICATION TO BAYESIAN NETWORK LEARNING

We ran our implementation of order-MCMC on four datasets available from the UCI repository (Blake and Merz, 1998). Votes2 was obtained by concatenating two random permutations of the 17-variable Votes

Table 2: Datasets, Generative Bayesian Network Models, and Parameter Settings Used in the Experiments for Bayesian Network Learning by Order-MCMC.

Name	n	#Samples	k	#Steps
Votes2	34	435	5	10000
Chess	37	3196	5	5000
10xPromoters	58	1060	4	2000
Splice	61	3190	4	2000
Alarm	37	50–5000	5	1000
Hailfinder	56	50–5000	4	1000

[1]The implementation will be made publicly available via the authors' home pages.

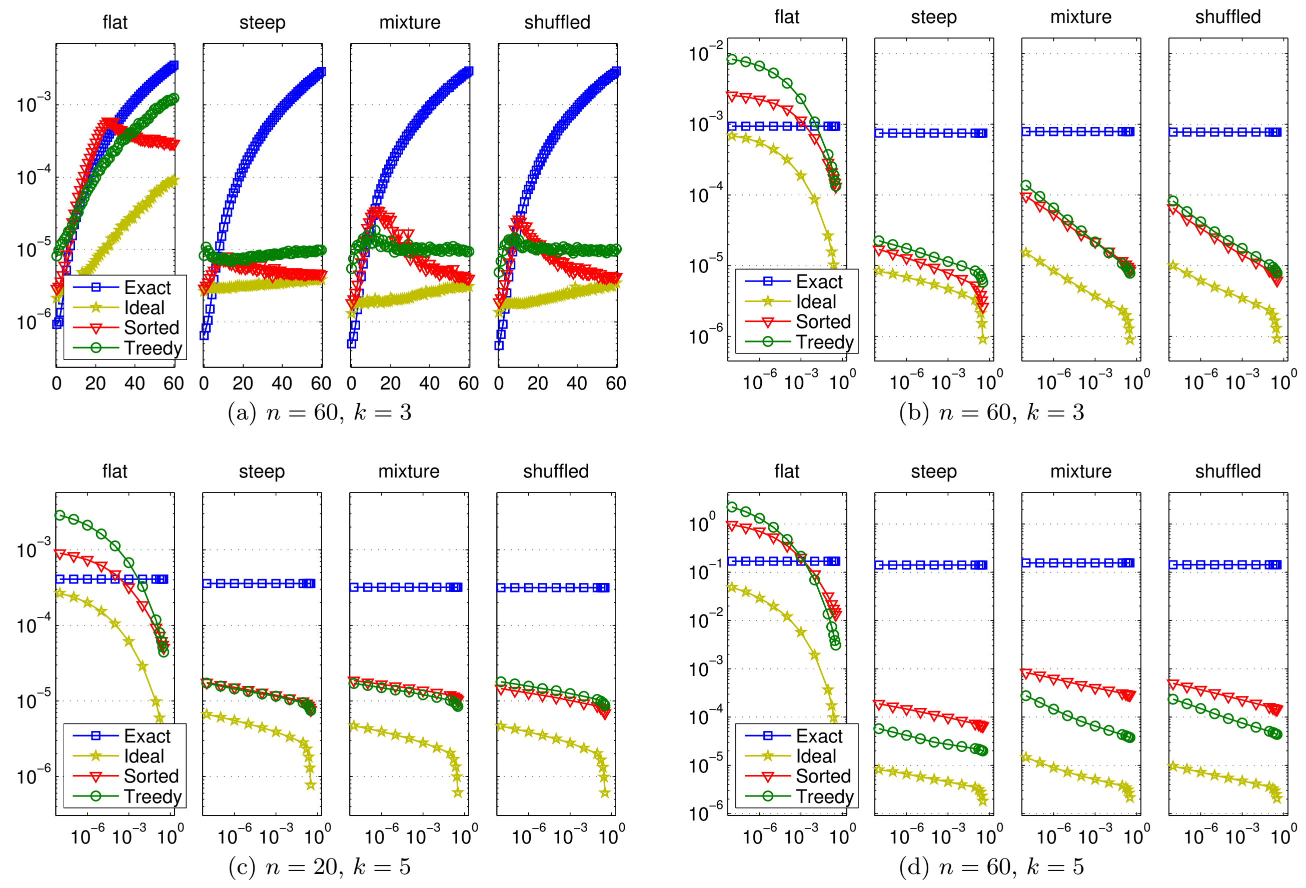


Figure 2: Runtime in artificial problem instances. The number of seconds per subset counting query for *Exact*, *Ideal*, *Sorted*, and *Treedy* are shown (a) as a function of the query set size, for a fixed approximation tolerance of 0.1, and (b, c, d) as a function of approximation tolerance.

dataset, so doubling the number variables. 10xPromoters was obtained by taking each sample of the Promoters dataset 10 times. The datasets Chess and Splice we used as such. In addition, we used datasets of varying sample sizes generated from the benchmark Bayesian network models Alarm (Beinlich et al., 1989) and Hailfinder (Abramson et al., 1996).[2] Table 2 shows the key parameters associated with the datasets and the order-MCMC method, including the maximum number of parents k and the number of MCMC steps. The posterior (the structure and parameter priors) was specified as in the experiments of Niinimäki et al. (2011).

We observe that approximation expedites the computations by one to several orders of magnitude; see Figure 3. As expected, the gain of approximation increases with larger datasets and larger error, being, however, significant already with 200 samples and easily tolerable error (say 1%). On the larger datasets *Treedy* performs consistently better than *Sorted*, the difference being sometimes nearly one order of magnitude (Chess and Alarm with an approximation tolerance of at least 1%).

Examining the effect of the query set size for the four datasets (Figure 3(a)) reveals that *Treedy* is consistently faster than *Sorted* on queries that are the hardest ones for *Sorted*. However, *Sorted* is typically faster than *Treedy* on the easier query sets. Thus it depends on the distribution of queries, whether *Treedy* or *Sorted* should be the algorithm of choice, or whether it would pay off to use the obvious hybrid: *Treedy* for smaller and *Sorted* for larger query sets.

The results for *Ideal* suggest that considerable further speedups, by one to two orders of magnitude, might be possible using still better algorithms and data structures.

[2]The datasets are available at `http://www.dsl-lab.org/supplements/mmhc_paper/mmhc_index.html`.

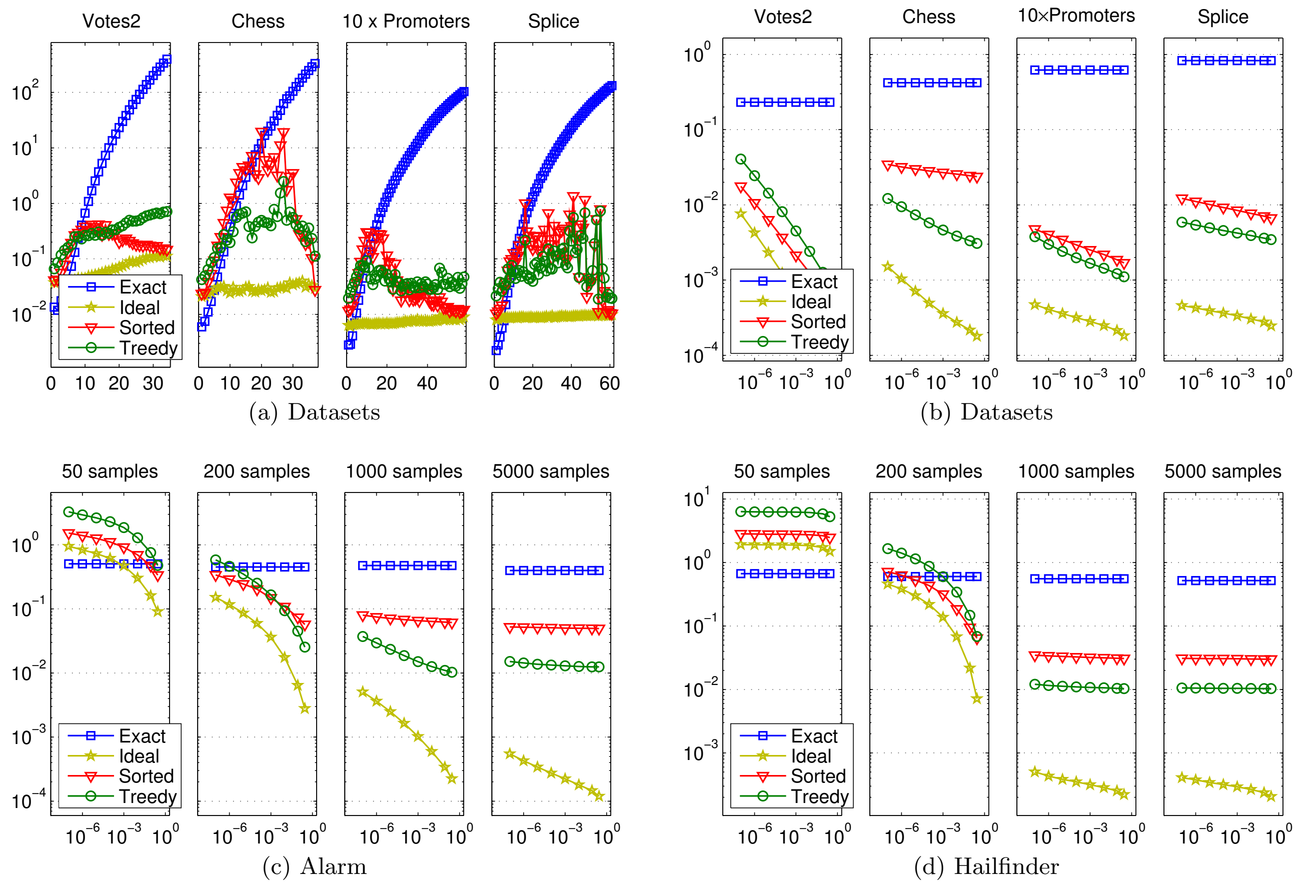


Figure 3: Runtime in the Bayesian network application. The number of seconds per MCMC step for *Exact*, *Ideal*, *Sorted*, and *Treedy* are shown (a) as a function of the query set size, for a fixed approximation tolerance of 0.1, and (b, c, d) as a function of approximation tolerance d. Since the approximated posterior probability is obtained as a product of n approximated total weights, the tolerance per total weight was set to d/n to guarantee the required accuracy.

5 DISCUSSION

We have studied approximation algorithms for subset counting and sampling queries. After observing how any collector algorithm for approximate counting queries can be turned into a sampling algorithm, we considered four collector algorithms. These include a slow exact algorithm that serves as a reference and an "ideal" one that we cannot implement efficiently but that provides us with a lower bound of the needed work. Our experimental results suggest that the two heuristic methods, *Sorted* and *Treedy*, can yield dramatic speed-ups (by several orders of magnitude) over the exact algorithm, while not quite achieving the ideal performance. Typically, *Treedy* performs as well as *Sorted* or significantly better.

We made the assumption that the given collection of subsets is downward closed. This assumption simplified the presentation and experimental settings. The assumption is, however, not well justified in general. Namely, the input collection can potentially be much smaller than its downward closure, in which case one could realistically hope for faster methods whose time requirement is determined by the size of the input rather than the size of the closure. We note that *Sorted* readily has this desirable property. For example, in the Bayesian network application it is quite plausible to expect that some potential parent sets have so small a weight that they can be discarded in the precomputation phase. Fortunately, the assumption of downward closedness seems not crucial for the validity of the presented methods. Indeed, the data structures and visiting orders underlying the methods *Exact* and *Treedy* can be pruned by introducing shortcuts. Using shortcuts, only subsets that belong to the collection need be visited, and so the other subsets can be discarded. We leave a more detailed description of this generalization and examination of its impact to the Bayesian network application to an extended version of this paper.

There are also other avenues for future research. We restricted our attention to methods whose memory requirement is roughly linear in the size of the input collection. It remains an open question, whether significant speedups can be achieved by investing somewhat more space. Likewise, we have only considered collector algorithms, and it is an open question, whether there exist faster algorithms of some different type.

Acknowledgements

The authors thank Petteri Kaski, Janne Korhonen, and Pekka Parviainen for valuable discussions about various aspects of the zeta transform, and the anonymous reviewers for useful comments on earlier versions of this paper. This research was supported in part by the Academy of Finland, Grants 125637, 218153, and 255675.

References

B. Abramson, J. Brown, W. Edwards, A. Murphy, and R. L. Winkler. Hailfinder: A Bayesian system for forecasting severe weather. *International Journal of Forecasting*, 12(1):57–71, 1996.

I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper. The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks. In J. Hunter, editor, *Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine*, pages 247–256. Springer-Verlag, 1989.

C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998.

W. Buntine. Theory refinement on Bayesian networks. In B. D'Ambrosio and P. Smets, editors, *UAI*, pages 52–60. Morgan Kaufmann, 1991.

M. Charikar, P. Indyk, and R. Panigrahy. New algorithms for subset query, partial match, orthogonal range searching, and related problems. In P. Widmayer, F. T. Ruiz, R. M. Bueno, M. Hennessy, S. Eidenbenz, and R. Conejo, editors, *ICALP*, volume 2380 of *Lecture Notes in Computer Science*, pages 451–462. Springer, 2002.

B. Ellis and W. H. Wong. Learning causal Bayesian network structures from experimental data. *Journal of the American Statistical Association*, 103:778–789, 2008.

N. Friedman and D. Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–125, 2003.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

R. Kennes. Computational aspects of the Moebius transformation of graphs. *IEEE Transactions on Systems, Man, and Cybernetics*, 22:201–223, 1991.

R. Kennes and P. Smets. Computational aspects of the Möbius transformation. In P. P. Bonissone, M. Henrion, L. N. Kanal, and J. F. Lemmer, editors, *UAI*, pages 401–416. Elsevier, 1990.

M. Koivisto. Advances in exact Bayesian structure discovery in Bayesian networks. In *UAI*, pages 241–248. AUAI, 2006.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

T. Niinimäki and M. Koivisto. Annealed importance sampling for structure learning in Bayesian networks. In *IJCAI*, 2013. To appear.

T. Niinimäki, P. Parviainen, and M. Koivisto. Partial order MCMC for structure discovery in Bayesian networks. In F. G. Cozman and A. Pfeffer, editors, *UAI*, pages 557–564. AUAI, 2011.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Francisco, 1988.

J. Pearl. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, Cambridge, 2000.

M. Pătraşcu. Unifying the landscape of cell-probe lower bounds. *SIAM J. Comput.*, 40(3):827–847, 2011.

F. Yates. The design and analysis of factorial experiments. *Harpenden: Imperial Bureau of Soil Science Technical Communication 35*, 1937.

Stochastic Rank Aggregation

Shuzi Niu **Yanyan Lan** **Jiafeng Guo** **Xueqi Cheng**
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, P. R. China
niushuzi@software.ict.ac.cn, {lanyanyan,guojiafeng,cxq}@ict.ac.cn

Abstract

This paper addresses the problem of rank aggregation, which aims to find a consensus ranking among multiple ranking inputs. Traditional rank aggregation methods are deterministic, and can be categorized into explicit and implicit methods depending on whether rank information is explicitly or implicitly utilized. Surprisingly, experimental results on real data sets show that explicit rank aggregation methods would not work as well as implicit methods, although rank information is critical for the task. Our analysis indicates that the major reason might be the unreliable rank information from incomplete ranking inputs. To solve this problem, we propose to incorporate uncertainty into rank aggregation and tackle the problem in both unsupervised and supervised scenario. We call this novel framework *stochastic rank aggregation* (St.Agg for short). Specifically, we introduce a prior distribution on ranks, and transform the ranking functions or objectives in traditional explicit methods to their expectations over this distribution. Our experiments on benchmark data sets show that the proposed St.Agg outperforms the baselines in both unsupervised and supervised scenarios.

1 INTRODUCTION

Rank aggregation is a central problem in many applications, such as metasearch, collaborative filtering and crowdsourcing tasks, where it attempts to find a consensus ranking among multiple ranking inputs.

In literature, deterministic rank information has been utilized to solve the rank aggregation problem, and we refer these methods as *deterministic rank aggregation.* The conventional methods can be further divided into two categories: explicit and implicit methods. For explicit methods (Aslam & Montague, 2001; Cormack et al., 2009), rank information is explicitly utilized for rank aggregation, e.g., (Aslam & Montague, 2001) uses the mean rank as the score and sorts items in ascending order. While for implicit methods (Gleich & Lim, 2011; Thurstone, 1927), rank information is used implicitly, e.g., (Thurstone, 1927) defines the generative probability of pairwise preferences based on rank information and adopts the maximum likelihood procedure for aggregation.

Although rank information is crucial for the rank aggregation task, surprisingly, the explicit methods would not work well as implicit methods in practice. Our experiments on real data sets show that the implicit methods outperform explicit methods, in both unsupervised and supervised scenarios.

The root of the problem may lie in the unreliable rank information from multiple incomplete ranking inputs. Typically, the ranking inputs in rank aggregation are partial rankings. For example, in metasearch, only top search results are returned from each meta search engine with respect to its repository; in a recommender system, users only rate the items they have ever seen. The incompleteness of the partial ranking makes the ranks of items no longer reliable. Taking the recommender system as an example, we do not know whether an unseen item will be ranked higher or lower than the already rated ones for the user. As indicated by (Voorhees, 2002; Farah & Vanderpooten, 2007), the incorrect rank information will reduce the performance of any explicit method. Therefore, it is not surprising to see the failure of explicit methods based on the current unreliable rank information.

To amend this problem, we propose to incorporate uncertainty into rank aggregation and tackle the problem in both unsupervised and supervised scenario. We refer this novel rank aggregation framework as *stochastic rank aggregation* (St.Agg for short). Specifically, a prior distribution derived from pairwise contests is introduced on ranks to accommodate the unreliable rank information, since it has been proven that rank information encoded in a pairwise way will be robust (Farah & Vanderpooten, 2007). We then define the new ranking functions or objectives (in unsupervised or supervised respectively) as the expectation of those in traditional methods with respect to the rank distribution. In unsupervised scenario, the new ranking functions are used to find the final ranking list. In supervised scenario, the learning to rank technique is employed to complete the rank aggregation task. Specifically, a feature representation for each item is first designed, based on explicit features in (Aslam & Montague, 2001) and latent features in (Volkovs et al., 2012; Kolda & Bader, 2009). A gradient method is then employed to optimize the new objectives.

Our experiments on benchmark data sets show that the proposed St.Agg significantly outperforms traditional rank aggregation methods, in both unsupervised and supervised scenarios. Furthermore, we conduct experiments to demonstrate that St.Agg is more robust than traditional methods, showing the benefit of uncertainty.

In summary, we propose a novel rank aggregation framework which incorporates uncertainty of rank information. Our major contributions include: (1) the finding that partial ranking inputs in rank aggregation will make the explicit methods fail, due to unreliable rank information; (2) the definition of rank distribution based on pairwise contests, which is the basis of stochastic rank aggregation; and (3) the proposal of a unified rank aggregation framework in both unsupervised and supervised scenarios.

The rest of the paper is organized as follows. We first introduce some backgrounds on rank aggregation in Section 2, and then conduct some empirical study and analysis on why explicit methods will not work well in rank aggregation. Section 4 describes the framework of stochastic rank aggregation, including the definition of rank distribution and St.Agg in both unsupervised and supervised scenarios. Section 5 presents our experimental results and Section 6 concludes the paper.

2 BACKGROUNDS

In this section, we introduce some backgrounds on rank aggregation, including problem definition, previous methods and evaluation measures.

2.1 PROBLEM DEFINITION

In unsupervised scenario, we are given a set of n items $\{x_1, \cdots, x_n\}$ and multiple ranking inputs $\tau_1, \cdots, \tau_m$ over these items. $\tau_i(x_j)$ stands for the position of x_j in the ranking τ_i. If the length of τ_i is n, τ_i is called a full ranking; otherwise, it is called a partial ranking. The goal of unsupervised rank aggregation is to find a final ranking $\pi \in \Pi$ over all the n items which best reflects the ranking order in the ranking inputs, where Π is the space of all the full ranking with length n. To achieve this goal, many aggregation algorithms try to optimize a similarity function F between the ranking inputs $\tau_1, \cdots, \tau_m$ and the final ranking result π, while some other aggregation algorithms directly give the form of their final ranking function f without explicit optimization objectives.

In supervised scenario, we are given N sets of items, denoted as $D_i = \{x_1^{(i)}, \cdots, x_{n_i}^{(i)}\}, i = 1, \cdots, N$. For each item set D_i, a collection of ranking inputs $\tau_1^{(i)}, \cdots, \tau_{m_i}^{(i)}$ are over this set. The ground-truth labels for all items are provided in the form of multi-level ratings, such as three level ratings (2:highly relevant, 1:relevant, 0:irrelevant), denoted as $Y^{(i)} = (y_1^{(i)}, \cdots, y_{n_i}^{(i)})$. In the training process, an aggregation function f of ranking inputs is learned by optimizing a sum of a similarity function F on N sets, and F measures the similarity between these ranking inputs and the corresponding ground-truth ranking. For prediction, given any item set $D = \{x_1, \cdots, x_n\}$ and m ranking inputs $\tau_1, \cdots, \tau_m$ over this set, the final ranking π is computed by $\pi = f(\tau_1, \cdots, \tau_m)$.

2.2 METHODS

Previous rank aggregation methods can be divided into two categories according to the way that rank information is used: explicit and implicit rank aggregation methods. Explicit methods directly utilize rank information to define the ranking function or the objective function. While for implicit methods, other information such as pairwise preference or listwise ranking are first constructed based on the rank information, and then leveraged for rank aggregation.

2.2.1 Unsupervised Aggregation Methods

Firstly, let we introduce the two kinds of methods in unsupervised scenario, respectively.

Explicit Methods. In unsupervised scenario, explicit methods define the ranking function as the sum of utility functions of each items's rank information, and then sort the items in descending order. The formulation is described as follows.

$$f(x_j) = \sum_{i=1}^{m} u(x_j, \tau_i), \tag{1}$$

where $u(\cdot, \cdot)$ stands for the utility function. For example, (Aslam & Montague, 2001) used the mean position as the ranking function as shown in Eq.(2), and (Cormack et al., 2009) defined the reciprocal rank as the ranking function to further emphasize the top ranked items as shown in Eq.(3).

$$f(x_j) = \sum_{i=1}^{m} u(x_j, \tau_i) = \sum_{i=1}^{m} (n - \tau_i(x_j)), \qquad (2)$$

$$f(x_j) = \sum_{i=1}^{m} u(x_j, \tau_i) = \sum_{i=1}^{m} \frac{1}{C + \tau_i(x_j)}, \qquad (3)$$

where C is a constant, n_i is the length of τ_i and $\tau_i(x_j)$ in both Eq.(2) and Eq.(3) means the position of x_j in ranking τ_i.

Implicit Methods. (Dwork et al., 2001) used a Local Kemenization procedure to approximate an optimal solution to minimize Kendall's tau distance. (Gleich & Lim, 2011) defined the difference between the pairwise preference matrix from ranking input and the aggregated preference matrix, and adopted matrix factorization for optimization. (Thurstone, 1927) and (Volkovs & Zemel, 2012) defined the similarity measure to be the generative probability of pairwise preferences and then adopted the maximum likelihood procedure for aggregation. (Guiver & Snelson, 2009) and (Lebanon & Lafferty, 2002) defined the similarity measure to be the generative probability of each ranking list and optimized this similarity function by a maximum likelihood procedure.

2.2.2 Supervised Aggregation Methods

Secondly, we continue to introduce the two kinds of methods in supervised scenario, respectively.

Explicit Methods. For explicit methods in supervised scenario, features are first extracted from the ranking inputs, and then the rank information generated by the ranking function of these features are directly used in the objective function. After that, learning-to-rank technique are utilized for optimization. For example, (Volkovs et al., 2012) proposed to use evaluation measures as the objective function, such as NDCG (Normalized Discounted Cumulative Gain) (Järvelin & Kekäläinen, 2002), ERR (Expected Reciprocal Rank) (Chapelle et al., 2009) and RBP (Rank Biased Precision) (Moffat & Zobel, 2008). This approach is called RankAgg, and we will review their mathematical formulations in Section 2.3.

Implicit Methods. (Liu et al., 2007) proposed to minimize the number of disagreeing pairs between the aggregated ranking result and the ground-truth. (Volkovs & Zemel, 2012) heuristically computed the similarity between the ranking input and the ground-truth to obtain the expertise degree of the corresponding annotator. The learned weights are then used to aggregate the ranking lists for future data. (Qin et al., 2010) introduced coset-permutation distance into Plackett-Luce model for rank aggregation.

2.3 EVALUATION MEASURES

Rank information is explicitly used in evaluation measures for rank aggregation, such as NDCG (Järvelin & Kekäläinen, 2002), RBP (Moffat & Zobel, 2008) and ERR (Chapelle et al., 2009). We would like to express these measures as the sum of differences on each item's generated rank and ground-truth, as shown in the following equation.

$$Ev(\pi, Y) = \sum_{j=1}^{n} v(y_j, r_j), \qquad (4)$$

where Ev stands for any evaluation measure, $v(\cdot,\cdot)$ stands for the difference function. We give the exact forms of NDCG, RBP and ERR as follows.

$$NDCG(\pi, Y) = \sum_{j=1}^{n} \frac{g(y_j)D(r_j)}{DCG_{max}(n)}, \qquad (5)$$

where r_j stands for the rank of x_j in the final ranking π. $g(y_j)$ is the gain function with $g(y_j) = 2^{y_j} - 1$, $D(r_j)$ is the discount function with $D(r_j) = \frac{1}{log(1+r_j)}$, and $DCG_{max}(n)$ stands for the maximum of $\sum_{j=1}^{n} g(y_j)D(r_j)$ over Π.

$$ERR(\pi, Y) = \sum_{j=1}^{n} \frac{1}{r_j} P\{\text{users stop at } r_j\}, \qquad (6)$$

where the probability $P\{\text{users stop at } r_j\}$ is defined as

$$\prod_{i \in \{i|r_i < r_j\}} (1 - \frac{2^{y_i} - 1}{2^{y_{max}} - 1}) \frac{2^{y_j} - 1}{2^{y_{max}}},$$

where y_{max} is the maximum of the ground-truth label.

$$RBP(\pi, Y) = \sum_{j=1}^{n} (1-p) y_j p^{r_j - 1}, \qquad (7)$$

where $p \in [0, 1]$ is a constant value, for example 0.95 used in this paper.

3 MOTIVATION

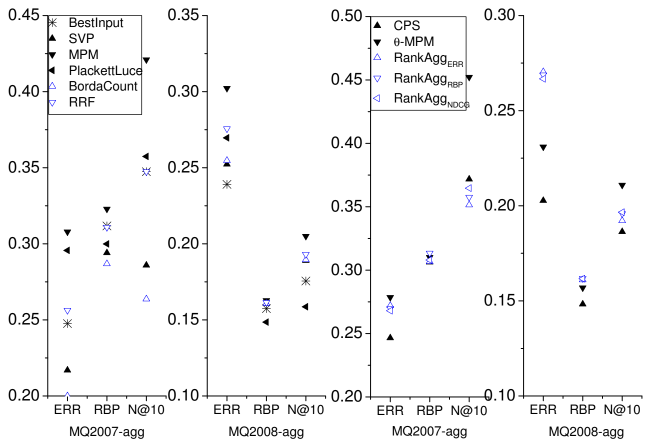
Rank information is crucial for rank aggregation, and evaluation measures in rank aggregation are often rank dependent. However, experimental results on real data sets show that performances of explicit rank aggregation methods cannot be comparable with implicit methods in most cases. Through analysis we find that the major reason lies in the unreliable rank information used in the explicit methods directly. This motivates us to design new aggregation methods to utilize rank information in a more robust way.

3.1 EMPIRICAL FINDINGS

Here we conduct experiments to compare performances between the explicit methods and the implicit methods on benchmark data sets MQ2007-agg and MQ2008-agg in LETOR4.0 in both unsupervised and supervised scenario. MQ2007-agg contains 1692 queries with 21 ranking inputs per query on average and MQ2008-agg contains 784 queries with 25 ranking inputs per query on average. In both data sets, three level relevance judgment per document is provided as the ground-truth.

In unsupervised scenario, we choose BordaCount (Aslam & Montague, 2001), RRF (Cormack et al., 2009) as the baselines of explicit method, and SVP (Gleich & Lim, 2011), MPM (Volkovs & Zemel, 2012) and PlackettLuce (Guiver & Snelson, 2009) as the baselines of implicit methods. In supervised scenario, we choose RankAgg (RankAgg$_{NDCG}$, RankAgg$_{ERR}$, RankAgg$_{RBP}$) (Volkovs et al., 2012) as the baselines of explicit methods, and CPS (Qin et al., 2010) and θ-MPM (Volkovs & Zemel, 2012) as the baselines of implicit methods, where the feature mapping method used in RankAgg in supervised scenario is Borda Count (Aslam & Montague, 2001).

The experimental results are shown in Figure 1, where Bestinput in Figure 1(a) stands for the method that directly utilizes the best ranking input in terms of evaluation measures as the output. It is obvious that implicit aggregation methods outperform explicit methods in most cases in both unsupervised scenario (Figure 1(a)) and supervised scenarios (Figure 1(b)), especially on MQ2007-agg.



(a) Unsupervised scenario (b) Supervised scenario

Figure 1: Performance Comparison between Existing Explicit and Implicit Methods

3.2 ANALYSIS

The contradiction between the experimental results and the intuition inspires us to revisit these explicit methods. The ranking inputs for aggregation, i.e. inputs in the data sets of MQ2007-agg and MQ2008-agg, are typically partial rankings. The incompleteness of the partial ranking makes the ranks of items no longer reliable, which causes the failure of these explicit methods according to (Voorhees, 2002; Farah & Vanderpooten, 2007).

Taking Borda Count and an example as follows to show the reason. $\tau_{1D} = a \succ b, \tau_{2D} = b \succ c, \tau_{3D} = c \succ d$ are three incomplete rankings over $D = \{a, b, c, d\}$, and the ground-truth ranking is $a \succ b \succ c \succ d$. Borda Count ranks the items by the mean positions of all the three lists. With assumption that $\tau_{1D}(a)=1, \tau_{1D}(b)=2, \tau_{2D}(b)=1, \tau_{2D}(c)=2, \tau_{3D}(c)=1, \tau_{3D}(d)=2$, the final ranking obtained from Borda Count in unsupervised scenario is $b \succ c \succ a \succ d$ or $c \succ b \succ a \succ d$, which is far from the ground-truth ranking. In supervised scenario, these unreliable rank positions used to calculate the Borda Count score are employed as features directly. These unreliable features lead to the failure of learning to rank algorithms.

Through further analysis, we find that the above failure is caused by unreliable rank information generated from partial ranking inputs. Specifically, the ranks of missing items are taken as NULL, which would not appear in the computation, and make the positions of remaining items no longer reliable. For example, c and d are missing in τ_{1D}, and thus the ranks of a and b are taken as 1 and 2, respectively. However, these ranks do not reflect the true absolute ranks of a and b if c and d are taken into consideration. In a word, the incomplete ranking inputs result in unreliable rank information, which leads to the failure of explicit aggregation methods. This motivates us to take the uncertainty of rank information into consideration and design new rank aggregation methods.

4 STOCHASTIC RANK AGGREGATION

Through analysis in section 3.2, it is important for aggregation methods to be able to accommodate the uncertainty of the rank information. To achieve this goal, we treat rank as a random variable and design a novel rank aggregation framework which transforms the ranking functions or objectives into the expectations over its distribution in both unsupervised and supervised scenario, namely St.Agg.

4.1 INCORPORATING UNCERTAINTY INTO RANKS

In section 3.1 we observe that these implicit methods with pairwise preferences as inputs outperform explicit methods in most cases. Inspired by this observation, we define the random variable of rank with respect to a ranking input τ as the result of pairwise contests, described as follows:

$$R(x_j,\tau)=\sum_{i=1,i\neq j}^{n} I(x_i \succ_\tau x_j), \tag{8}$$

where $R(x_j,\tau)$ is the rank of item x_j in τ, and $I(x_i \succ_\tau x_j)$ stands for the event that item x_j is beaten by item x_i. We assume that the pairwise contest between x_i and x_j follows a Bernoulli trail with probability $p(x_i \succ_\tau x_j)$ that x_j is beaten by x_i. Therefore, the rank $R(x_j,\tau)$ is a Binomial-like random variable, equal to the number of successes of $n-1$ Bernoulli trials.

Furthermore, the rank distribution for each item x_j can be computed from τ in the following process.

(1) If we only have item x_j, its rank will be 0 (the best rank). Thus the initialization distribution of item x_j is defined as:

$$P^{(1)}(R(x_j,\tau)=r)=\delta(r)=\begin{cases}1, & r=0\\ 0, & r>0\end{cases} \tag{9}$$

(2) Each time we add a new item x_i, the rank will get one larger if x_j is beaten by x_i and it will stay unchanged otherwise. Therefore the rank distribution for item x_j will be updated by the following recursive relation:

$$\begin{aligned}&P^{(t)}(R(x_j,\tau)=r)\\ &=P^{(t-1)}(R(x_j,\tau)=r-1)p(x_i \succ_\tau x_j)\\ &\quad+P^{(t-1)}(R(x_j,\tau)=r)(1-p(x_i \succ_\tau x_j)).\end{aligned} \tag{10}$$

(3) After $n-1$ iterations, $P^{(n)}(R(x_j,\tau)=r)$ will be the final distributions $P(R(x_j,\tau)=r)$.

Based on the definition of rank distribution above, the ranking function or objective function used in an explicit method can be turned into a new function by taking the expectation over this rank distribution, which will be used in the new rank aggregation method, described in the following section.

4.2 UNSUPERVISED ST.AGG

First we introduce the specific form of pairwise probability $p(x_i \succ_\tau x_j)$ for the computation of rank distribution $P(R(x_j,\tau)=r)$ in unsupervised scenario. The expectations of ranking functions in explicit methods are then computed over the rank distribution, and used to obtain the final aggregated ranking. We denote our unsupervised St.Agg as St.Agg$_f$ for different ranking functions.

4.2.1 Definition of Pairwise Probability

Inspired by the robust rank difference information (Farah & Vanderpooten, 2007) underlying in the ranking list, we define $p(x_i,x_j)$ as the function of rank difference between x_i and x_j in the full ranking τ_{D_τ} over the subset $D_\tau \subseteq D$ such as $\frac{|\tau_{D_\tau}(x_i)-\tau_{D_\tau}(x_j)|}{n}$. We would like to give an example of the definition of the pairwise probability $p(x_i \succ_\tau x_j)$ satisfying $p(x_i \succ_\tau x_j)+p(x_j \succ_\tau x_i)=1$ as below :

$$\begin{cases}\min\{p(x_i,x_j),1-p(x_i,x_j)\}, & \text{if } \tau_{D_\tau}(x_i)>\tau_{D_\tau}(x_j)\\ \max\{p(x_i,x_j),1-p(x_i,x_j)\}, & \text{if } \tau_{D_\tau}(x_i)<\tau_{D_\tau}(x_j)\\ 0.5, & \text{otherwise}\end{cases} \tag{11}$$

Incorporating Eq.(11) to the recursive process Eq.(10), we can obtain the probability distribution over ranks $P(R(x_j,\tau)=r)$.

4.2.2 Expectations

As mentioned in section 2, the ranking function $f(x_j)$ in explicit methods is a sum of the utility function of rank information of a certain item from multiple ranking inputs. Thus we can simply calculate the expectation of the ranking function $f_s(x_j)$ over the rank distribution proposed above.

For example, the new ranking function $f_s(x_j)$ in St.Agg$_{BC}$ and St.Agg$_{RRF}$ can be obtained by incorporating rank distribution $P(R(x_j,\tau_i)=r)$ into the mean position function in Eq.(2) and the reciprocal rank function in Eq.(3) respectively, as shown below.

$$f_s(x_j)=\frac{1}{m}\sum_{i=1}^{m}\sum_{r=0}^{n-1}(n-r)P(R(x_j,\tau_i)=r), \tag{12}$$

$$f_s(x_j)=\sum_{i=1}^{m}\sum_{r=0}^{n-1}\frac{P(R(x_j,\tau_i)=r)}{r+C}. \tag{13}$$

where BC is short for Borda Count. For such new aggregation methods, the final ranking is obtained by sorting in descending order of $f_s(x_j)$.

4.3 SUPERVISED ST.AGG

In supervised scenario, we utilize the state-of-the-art learning framework for the optimization problem in rank aggregation like RankAgg$_F$ (Volkovs et al., 2012) mentioned in the section 2.2. Similarly we denote our supervised St.Agg as St.Agg$_F$ for different definitions of optimization functions.

Firstly, the specific form of pairwise probability $p(x_i \succ_\pi x_j)$ for the computation of rank distribution $P(R(x_j,\pi)=r)$ is defined based on the aggregated ranking π. Then the optimization objectives in our supervised St.Agg can be defined as the expectation of these objectives over rank distributions $P(R(x_j,\pi)=r)$. To solve this aggregation problem by a learning procedure, a proper feature mapping is first designed for representation, and then a gradient-based optimization method is adopted to learn the ranking function f.

4.3.1 Definition of the Pairwise Probability

Pairwise probability is defined on the basis of the ranking function f. Therefore, we define the score of each item x_i as a normal random variable denoted as s_i with expectation $f(x_i)$ and variance σ^2, i.e. $s_i \sim \mathcal{N}(f(x_i), \sigma^2)$.

Similar to the definition in unsupervised scenario, the pairwise probability can be defined as the function of score difference to reflect that the larger the score difference between x_i and x_j, the more probable that x_j is defeated by x_i. Thus $p(x_i \succ_\pi x_j)$ can be defined as $P(s_i - s_j > 0)$, where $s_i - s_j \sim \mathcal{N}(f(x_i) - f(x_j), 2\sigma^2)$. The computation of $p(x_i \succ_\pi x_j)$ can be taken as the following Eq.(14).

$$p(x_i \succ_\pi x_j) = \int_0^{+\infty} \frac{1}{2\sigma\sqrt{\pi}} e^{-\frac{(x-(f(x_i)-f(x_j)))^2}{4\sigma^2}} dx \qquad (14)$$

Applying the specific form of $p(x_i \succ_\pi x_j)$ in Eq.(14) into the recursive computation of $P(R(x_j,\pi)=r)$ in Eq.(10), the Binomial-like distribution $P(R(x_j,\pi)=r)$ are computed. For the convenience of computing the derivation, the Binomial-like distribution can be approximated by the normal distribution with means $\sum_{i=1, i\neq j}^n p(x_i \succ_\pi x_j)$ and variance $\sum_{i=1,i\neq j}^n p(x_i \succ_\pi x_j)(1-p(x_i \succ_\pi x_j))$.

4.3.2 Expectations

Objective functions in explicit methods mentioned in section 2.2 can be expressed as the sum of difference functions, such as ERR in Eq.(6), RBP in Eq.(7) and NDCG in Eq.(5). Incorporating $P(R(x_j,\pi)=r)$ into these objectives, the expectation of them can be easily obtained by taking the expectation of the difference functions $v(\cdot,\cdot)$ over the rank distribution, denoted as ERR$_s$, RBP$_s$ and NDCG$_s$. The general form $Ev_s(\pi, Y)$ and these three measures are listed below.

$$Ev_s(\pi, Y) = \sum_{j=1}^{n}\sum_{r=0}^{n-1} v(y_i, r) P(R(x_j,\pi)=r),$$

$$\mathrm{ERR_s}(\pi, Y) = \sum_{j=1}^{n}\sum_{r=0}^{n-1} \frac{P(\text{users stop at } r) P(R(x_j,\pi)=r)}{r+1},$$

$$\mathrm{RBP_s}(\pi, Y) = (1-p)\sum_{j=1}^{n}\sum_{r=0}^{n-1} y_j p^r P(R(x_j,\pi)=r),$$

$$\mathrm{NDCG_s}(\pi, Y) = \frac{\sum_{j=1}^{n}\sum_{r=0}^{n-1} g(y_j) D(1+r) P(R(x_j,\pi)=r))}{DCG_{max}(n)}.$$

4.3.3 Feature-based Learning Framework

The remaining question is how to optimize these expected objectives in a feature-based learning framework. Stage I is to design a better feature mapping for item representation; Stage II is the learning process of the supervised St.Agg, i.e. St.Agg$_F$.

Stage I: Feature mapping. In literature, feature mapping techniques for rank aggregation can be classified into three groups in terms of information used in the feature extraction process.

(1)*Features in terms of user-item relation.* Borda Count is such a natural feature, which aims to assign an item a relevance score per ranking input according to its position in this ranking input only. For example $nBC(x_i, \tau_i) = \frac{n-\tau_i(x_i)}{n}$, so $\Psi_{BF}(i) = [nBC(x_i,\tau_1), \cdots, nBC(x_i,\tau_m)]$.

(2)*Features in terms of both user-item and item-item relations.* Maksims et al. (Volkovs et al., 2012) proposed a transformation from all the ranking inputs into latent feature representations for each item based on SVD factorization. Each ranking input τ_i can be transformed into a pairwise preference based matrix denoted as P_i. Each matrix P_i can be approximated by rank-p singular vector decomposition $P_i \approx U_i \Sigma_i V_i'$. Therefore, each item x_i can be represented as a SVD-based feature vector from m ranking inputs, $\Psi_{MF}(i) = [U_1(i,:), diag(\Sigma_1), V_1(i,:), \cdots, U_m(i,:), diag(\Sigma_m), V_m(i,:)]$.

(3)*Features in terms of all of the three relations.* Tensor factorization method can take the item-item, item-user, user-user relations into consideration (Hong et al., 2012). In this paper, we use CanDeComp/Parafac (CP) decomposition (Kolda & Bader, 2009) for tensor factorization due to the nice property that it has a unique solution of decomposition, which provides a theoretical guarantee to get a stable solution. Specifically, the item-item-user tensor T with $T(:,:,i) = P_i$ is factorized as $T = \sum_{j=1}^{p} \lambda_j U_{:,j} V_{:,j} W_{:,j}$. Therefore CP-based feature vector for item x_i is represented as $\Psi_{TF}(i) = [U(i,:), V(i,:)]$.

Stage II: Gradient-based learning algorithm. Suppose f is a linear model with parameter w. Here we use gradient method for the optimization of these expected objectives such as ERR$_s$, RBP$_s$ and NDCG$_s$. The gradients of these expected objectives are computed as below.

$$\sum_{j=1}^{n}\sum_{r=0}^{n-1}\frac{\partial Ev_s(\pi, Y)}{\partial P(R(x_j,\pi)=r)}\frac{\partial P(R(x_j,\pi)=r)}{\partial w} \tag{15}$$

For ERR$_s$, RBP$_s$ and NDCG$_s$, the only difference of their partial derivatives lies in the first part of derivatives in the Eq.(15), which can be easily derived as follows.

$$\frac{\partial \mathrm{ERR_s}(\pi, Y)}{\partial P(R(x_j,\pi)=r)} = \frac{P(\text{users stop at } r)}{r+1},$$

$$\frac{\partial \mathrm{RBP_s}(\pi, Y)}{\partial P(R(x_j,\pi)=r)} = (1-p)y_j p^r,$$

$$\frac{\partial \mathrm{NDCG_s}(\pi, Y)}{\partial P(R(x_j,\pi)=r)} = \frac{g(y_j)D(r)}{DCG_{max}(n)}.$$

5 EXPERIMENTS

In this section we compare the performance of our aggregation methods St.Agg with traditional methods in terms of ERR, RBP and NDCG on two benchmark aggregation data sets, i.e. MQ2007-agg and MQ2008-agg in LETOR4.0. In unsupervised scenario, the ground-truth is only used for evaluation; In supervised scenario, the ground-truth is employed for both training and evaluation. Finally we make a detailed analysis on the robustness of St.Agg.

5.1 EFFECTIVENESS OF UNSUPERVISED ST.AGG

As summarised in Section 2, the baseline methods fall into two groups in unsupervised scenario, i.e. the implicit group including Markov Chain based methods denoted as MCLK (Dwork et al., 2001), SVP (Gleich & Lim, 2011), MPM (Volkovs & Zemel, 2012) and Plackett-Luce (Guiver & Snelson, 2009), and the explicit group including Borda Count (Aslam & Montague, 2001) and RRF (Cormack et al., 2009). We implement two unsupervised St.Agg methods including St.Agg$_{BC}$ and St.Agg$_{RRF}$.

We use the standard partition in LETOR4.0. For parameter setting, we choose the parameters when a method reaches its best performance on validation set. For example, parameter C of RRF is set to 40 on MQ2007-agg and 10 on MQ2008-agg. The learning rate is 0.1 and precision is 0.01 for SVP on both data sets.

The experimental results are shown in Table 1. Firstly, we make a comparison between the explicit methods and St.Agg in terms of NDCG@5, NDCG@10, ERR and RBP. We can see that St.Agg$_{BC}$ and St.Agg$_{RRF}$ have obvious advantage over the explicit methods Borda Count and RRF in terms of all the measures. For example, the highest performance improvement of St.Agg$_{BC}$ is 79.7% in terms of NDCG@5 on MQ2007-agg compared with Borda Count. The highest performance improvement of St.Agg$_{RRF}$ is 32.3% in terms of NDCG@5 compared with RRF. Similar results can be found on MQ2008-agg. It demonstrates that explicit rank aggregation methods can be significantly improved by incorporating uncertainty into rank aggregation.

Secondly, we make a comparison between the implicit methods and St.Agg in terms of NDCG@5, NDCG@10, ERR and RBP. We can see that St.Agg is consistently better even than the best implicit method (MPM). Compared with MPM, St.Agg$_{RRF}$ achieves 5.2% higher in terms of NDCG@5 on MQ2007-agg. In terms of NDCG@10, our St.Agg$_{BC}$ performs 4.8% better than the best implicit method (MPM) on MQ2007-agg. Similar results can be found on the MQ2008-agg. Therefore, we conclude that explicit rank aggregation methods can outperform the implicit methods after incorporating uncertainty into rank aggregation.

In summary, St.Agg$_f$ with expected ranking function can improve the performance compared with the explicit methods which utilize rank information directly for aggregation. It also turned out to be better than the state-of-art implicit aggregation methods on both data sets in terms of all the evaluation measures. Therefore, our proposal to incorporate uncertainty into rank aggregation, i.e. stochastic rank aggregation, is significant for this task.

5.2 EFFECTIVENESS OF SUPERVISED ST.AGG

Similarly in supervised scenario, our baseline methods fall into two categories: (1) implicit rank aggregation methods including CPS (Qin et al., 2010) and θ-MPM (Volkovs & Zemel, 2012); and (2) explicit rank aggregation methods including methods mentioned in (Volkovs et al., 2012), denoted as RankAgg.

Both RankAgg and St.Agg are in a feature-based learning framework. Therefore, it is also a key problem to design a feature mapping for each item. In this paper, three mappings are adopted: (1) Borda Feature Ψ_{BF}; (2) SVD-based Features Ψ_{MF}; and (3) CP-based Features Ψ_{TF}. RankAgg and St.Agg under these different mappings are denoted as RankAgg(Ψ_{BF}), RankAgg(Ψ_{MF}), RankAgg(Ψ_{TF}) and St.Agg(Ψ_{BF}), St.Agg(Ψ_{MF}) St.Agg(Ψ_{TF}), respectively.

We use the standard partition in LETOR4.0, and em-

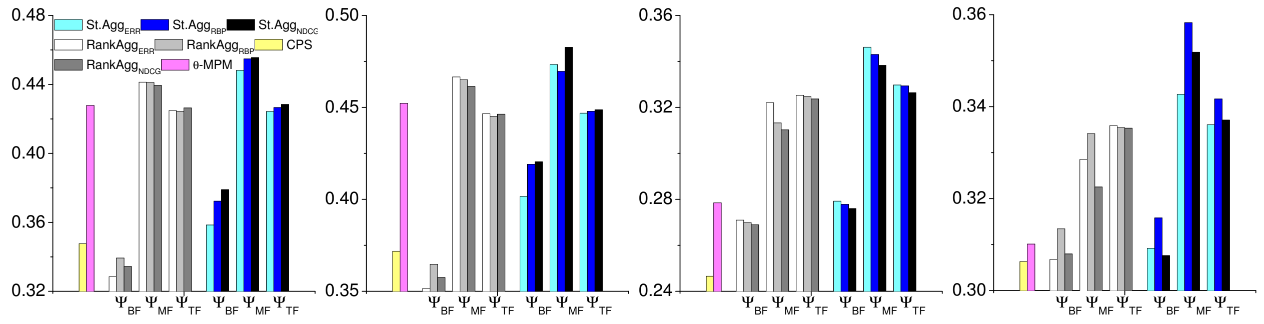
Table 1: Performance Comparison under Unsupervised Methods on MQ2007-agg and MQ2008-agg. All the results with bold type are significantly better than the baseline methods (p-value< 0.05).

(a) MQ2007-agg

Methods	N@5	N@10	ERR	RBP
BestInput	0.3158	0.3474	0.2476	0.3117
MCLK	0.2098	0.2450	0.1905	0.2798
SVP	0.2582	0.2859	0.2169	0.2941
Plackett-Luce	0.3462	0.3574	0.2957	0.2999
MPM	0.3986	0.4210	0.3078	0.3229
BordaCount	0.2325	0.2637	0.2000	0.2868
RRF	0.3172	0.3474	0.2563	0.3108
St.Agg$_{BC}$	0.4179	0.4384	0.3197	**0.3347**
St.Agg$_{RRF}$	**0.4195**	**0.4404**	**0.3199**	0.3346

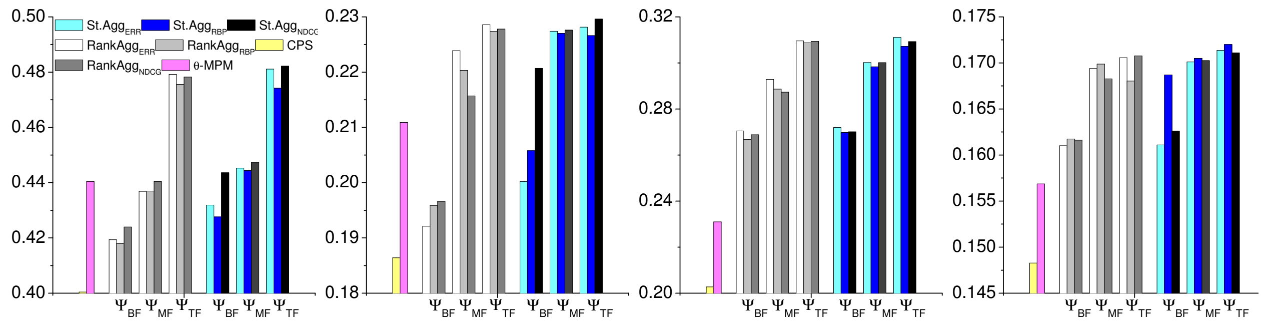
(b) MQ2008-agg

Methods	N@5	N@10	ERR	RBP
BestInput	0.3813	0.1756	0.2391	0.1574
MCLK	0.3402	0.1431	0.2055	0.1449
SVP	0.4004	0.1890	0.2523	0.1606
Plackett-Luce	0.3737	0.1586	0.2696	0.1485
MPM	0.4283	0.2050	0.3023	0.1628
BordaCount	0.4052	0.1895	0.2547	0.1607
RRF	0.4239	0.1931	0.2756	0.1615
St.Agg$_{BC}$	**0.4515**	0.2151	0.3021	0.1671
St.Agg$_{RRF}$	0.4512	**0.2157**	**0.3028**	**0.1673**



(a) NDCG@5 (b) NDCG@10 (c) ERR (d) RBP

Figure 2: Performance Comparison of Supervised Aggregation Methods on MQ2007-agg



(a) NDCG@5 (b) NDCG@10 (c) ERR (d) RBP

Figure 3: Performance Comparison of Supervised Aggregation Methods on MQ2008-agg

ploy five-fold cross validation to evaluate the performance of each methods. For gradient descent procedure in CPS (Qin et al., 2010), RankAgg (Volkovs et al., 2012) and our St.Agg, learning rate is chosen from 10^{-1} to 10^{-6} when the best performance is achieved on the validation set with the maximal number of iterations is 500. An additional parameter σ needs to be tuned from 10^{-1} to 10^{-4} for St.Agg.

5.2.1 Comparison of Aggregation Methods

We first compare the performances of different methods, and the experimental results are listed in Figure 2 and Figure 3. We can see that St.Agg outperforms RankAgg consistently in terms of NDCG@10, ERR and RBP under each feature mapping (p-value< 0.05). For example in Figure 2 on MQ2007-agg under Ψ_{MF}, the improvement of St.Agg$_{NDCG}$ over RankAgg$_{NDCG}$

is 4.6% in terms of NDCG@10, the improvement of St.Agg$_{ERR}$ over RankAgg$_{ERR}$ is 7.5% in terms of ERR, and the improvement of St.Agg$_{RBP}$ over RankAgg$_{RBP}$ is 7.2% in terms of RBP.

Compared with the best of implicit aggregation methods (θ-MPM) in Figure 2 on MQ2007-agg, St.Agg performs consistently better under any feature mapping (p-value< 0.01). Specifically, the improvement of St.Agg$_{NDCG}(\Psi_{MF})$ over θ-MPM is 6.73% in terms of NDCG@10, the improvement of St.Agg$_{ERR}(\Psi_{MF})$ over θ-MPM is 24.3% in terms of ERR and the improvement of St.Agg$_{RBP}(\Psi_{MF})$ over θ-MPM is 15.5% in terms of RBP. Similar results can be found on MQ2008-agg as shown in Figure 3.

To sum up, we can see that our proposed St.Agg can significantly outperform all these baselines in terms of NDCG, ERR and RBP.

5.2.2 Feature Mapping Comparison

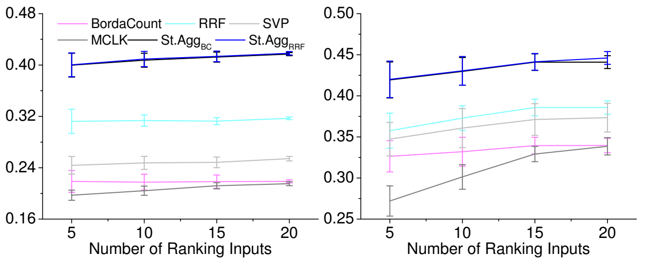
We further compare the effectiveness of different feature mappings. From the results in Figure 2, we can see that Ψ_{MF} is the best among all the three mappings on MQ2007-agg (p-value< 0.01). For example, performances on Ψ_{MF} are consistently better than the other two mappings Ψ_{BF} and Ψ_{TF} for both St.Agg$_{NDCG}$ and RankAgg$_{ERR}$. In Figure 3, obviously Ψ_{TF} is the best among all the three mappings on MQ2008-agg (p-value< 0.01). For example, performances on Ψ_{TF} are consistently better than that on Ψ_{MF} and Ψ_{BF} for both St.Agg$_{NDCG}$ and RankAgg$_{NDCG}$.

Through above analysis, Ψ_{MF} is best on MQ2007-agg and Ψ_{TF} is the best on MQ2008-agg. Since MQ2008-agg is much smaller and noisier than MQ2007-agg, our experimental results agreed with the previous findings that feature mappings based on tensor factorization will be more robust for sparsity and noise (Kolda & Bader, 2009).

5.3 ROBUSTNESS ANALYSIS OF ST.AGG

It is important to consider the robustness of rank aggregation methods. Here robustness means that the comparative performance will change little along with different ranking inputs, as defined in (Carterette & Petkova, 2006). Considering the computational efficiency, here we only take unsupervised St.Agg for example. With the number of ranking inputs from 5 to 20 with a step 5, we randomly choose the ranking inputs from the whole data sets 20 times. Each point in Figure 4 depicts the average NDCG@5 obtained on these 20 results.

It is obvious that NDCG@5 of St.Agg keeps high above all these explicit and implicit methods as the number



(a) MQ2007-agg (b) MQ2008-agg

Figure 4: Robustness Analysis of St.Agg

of ranking inputs increases in the data set. Therefore, St.Agg is very robust to the change of ranking inputs by incorporating uncertainty into ranks. In addition, the performance of each method tends to be stable with the increase of the number of ranking inputs since its variance is smaller and smaller.

6 CONCLUSION

In this paper, we propose a novel rank aggregation framework to incorporate uncertainty to this task, namely stochastic rank aggregation (i.e. St.Agg).

We give some empirical results and analysis to show that unreliable rank information from incomplete ranking inputs will make the approaches directly using rank information fail in practice. To tackle this problem, we propose to treat rank as a random variable and define the distribution by pairwise contests. After that, a novel rank aggregation framework in both unsupervised and supervised scenario is proposed, which takes the expectations of traditional ranking functions or objective functions for optimization. Finally, our extensive experiments on benchmark data sets show that the proposed St.Agg is better in terms of both effectiveness and robustness.

For future work, it is interesting to investigate how to incorporate uncertainty to implicit methods, and whether there are better ways to define the rank distribution.

Acknowledgements

This research work was funded by the National Natural Science Foundation of China under Grant No. 61232010, No. 61203298, No. 61003166 and No. 60933005, 863 Program of China under Grants No. 2012AA011003, and National Key Technology R&D Program under Grant No. 2011BAH11B02, No. 2012BAH39B02, No. 2012BAH39B04. We also wish to express our thanks to Yahoo!Research for providing the Yahoo!Rand dataset to us.

References

Aslam, J. A., & Montague, M. (2001). Models for metasearch. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval SIGIR2001* (pp. 276–284). New Orleans, Louisiana, United States: ACM.

Carterette, B., & Petkova, D. (2006). Learning a ranking from pairwise preferences. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval SIGIR2006* (pp. 629–630). Seattle, Washington, USA: ACM.

Chapelle, O., Metlzer, D., Zhang, Y., & Grinspan, P. (2009). Expected reciprocal rank for graded relevance. *Proceedings of the 18th ACM international conference on Information and knowledge management CIKM2009* (pp. 621–630). Hong Kong, China.

Cormack, G. V., Clarke, C. L. A., & Buettcher, S. (2009). Reciprocal rank fusion outperforms condorcet and individual rank learning methods. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval SIGIR2009* (pp. 758–759). New York, NY, USA: ACM.

Dwork, C., Kumar, R., Naor, M., & Sivakumar, D. (2001). Rank aggregation methods for the web. *Proceedings of the 10th international conference on World Wide Web WWW2001* (pp. 613–622). New York, NY, USA: ACM.

Farah, M., & Vanderpooten, D. (2007). An outranking approach for rank aggregation in information retrieval. *Proceedings of the 30th international ACM SIGIR conference on Research and development in information retrieval SIGIR2007* (pp. 591–598). Amsterdam, The Netherlands: ACM.

Gleich, D. F., & Lim, L.-h. (2011). Rank aggregation via nuclear norm minimization. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining KDD2011* (pp. 60–68). San Diego, California, USA: ACM.

Guiver, J., & Snelson, E. (2009). Bayesian inference for plackett-luce ranking models. *Proceedings of the 26th Annual International Conference on Machine Learning ICML2009* (pp. 377–384). New York, NY, USA: ACM.

Hong, L., Bekkerman, R., Adler, J., & Davison, B. D. (2012). Learning to rank social update streams. *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval SIGIR2012* (pp. 651–660). Portland, Oregon, USA: ACM.

Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, *20*, 422–446.

Kolda, T. G., & Bader, B. W. (2009). Tensor decompositions and applications. *SIAM Rev.*, *51*, 455–500.

Lebanon, G., & Lafferty, J. D. (2002). Cranking: Combining rankings using conditional probability models on permutations. *roceedings of the 19th Annual International Conference on Machine Learning ICML2002* (pp. 363–370).

Liu, Y.-T., Liu, T.-Y., Qin, T., Ma, Z.-M., & Li, H. (2007). Supervised rank aggregation. *Proceedings of the 16th international conference on World Wide Web WWW2007* (pp. 481–490). New York, NY, USA: ACM.

Moffat, A., & Zobel, J. (2008). Rank-biased precision for measurement of retrieval effectiveness. *ACM Trans. Inf. Syst.*, *27*, 2:1–2:27.

Qin, T., Geng, X., & Liu, T.-Y. (2010). A new probabilistic model for rank aggregation. *Advances in Neural Information Processing Systems 23 NIPS2010* (pp. 1948–1956).

Thurstone, L. L. (1927). The method of paired comparisons for social values. *The Journal of Abnormal and Social Psychology*, *21*, 384–400.

Volkovs, M. N., Larochelle, H., & Zemel, R. S. (2012). Learning to rank by aggregating expert preferences. *Proceedings of the 21st ACM international conference on Information and knowledge management CIKM2012* (pp. 843–851). Maui, Hawaii, USA: ACM.

Volkovs, M. N., & Zemel, R. S. (2012). A flexible generative model for preference aggregation. *Proceedings of the 21st international conference on World Wide Web WWW2012* (pp. 479–488). New York, NY, USA: ACM.

Voorhees, E. M. (2002). The philosophy of information retrieval evaluation. *CLEF '01* (pp. 355–370). Springer-Verlag.

Pay or Play

Sigal Oren *
Cornell University

Michael Schapira
Hebrew University
and Microsoft Research

Moshe Tennenholtz
Technion - Israel Institute of Technology
and Microsoft Research

Abstract

We introduce the class of *pay or play* games, which captures scenarios in which each decision maker is faced with a choice between two actions: one with a fixed payoff and another with a payoff dependent on others' selected actions. This is, arguably, the simplest setting that models selection among certain and uncertain outcomes in a multi-agent system. We study the properties of equilibria in such games from both a game-theoretic perspective and a computational perspective. Our main positive result establishes the existence of a semi-strong equilibrium in every such game. We show that although simple, pay or play games contain well-studied environments, e.g., vaccination games. We discuss the interesting implications of our results for these environments.

1 Introduction

The situation in which a decision-maker has to choose between an action with fixed, certain, outcome to a course of action with uncertain consequences is a fundamental topic in decision making under uncertainty. We introduce a new framework, called *pay or play.* In pay or play each of multiple decision makers must choose among an action with a known, fixed, payoff, and an action interpreted as participation in a game with other decision makers. The outcome of this game is dependent on who of the other decision makers also choose to take part in this game. The pay or play setting captures what is arguably the simplest scenario in which decision makers select between certain and uncertain outcomes, and the realization of the uncertain outcome is solely dependent on the decision makers and not on "nature". Importantly, in addition to its theoretical and conceptual appeal, pay or play encompasses, unifies, and abstracts classical models of immunization and of differential pricing.

A Game-Theoretic Formulation. We now give an informal, high-level, exposition of our (game-theoretic) pay or play model. In a pay or play game there are n self-interested players, each with two possible strategies (actions). Each player i has a cost function c_i which specifies, for every n-tuple of players' strategies, the cost of player i. c_i is such that whenever player i's strategy is pay his cost is some fixed value h_i, regardless of what the other players' strategies are. When player i's strategy is play, however, his cost is a function of the other players whose strategy is also play. We require each cost function c_i to be monotone nondecreasing, i.e., as more players choose play the cost of player i cannot decrease.

We are interested in the properties of (Nash) equilibria in this game-theoretic setting. An equilibrium is an n-tuple of strategies from which no player wishes to unilaterally deviate. We explore both pure (deterministic) equilibria, in which each player must choose one of these two strategies, and mixed (randomized) equilibria in which a player can choose a probability distribution over the two strategies. We tackle fundamental questions, including: Does a pure equilibrium always exist? Are equilibria in this environment "globally efficient"? What is the complexity of determining the existence and computing equilibria? And more.

Our Contributions. We study the properties of equilibria in pay or play games both from a game-theoretic perspective and a computational perspective. We now briefly summarize our results:

We begin by showing that a pure Nash equilibrium may not always exists and characterize some subclasses pay or play games which always admit a pure Nash equilibrium. The next natural question is how

* This work was done while S. Oren was a research intern at Microsoft Research .

hard is it to determine whether such an equilibrium exists or not—a question tackled in a large variety of other game-theoretic contexts. We show that this task is, in general, intractable from both a computational perspective (NP-hard) and information-theoretic (communication complexity) perspective.

A main criticism against Nash equilibria is that they are not resilient to deviations by coalitions of players. Equilibria that are resilient against all such deviations, called "strong equilibria", are hence of special interest. We identify conditions for the existence of a strong equilibrium. Our main positive result is that *any* pay or play game admits an equilibrium with a slightly weaker property, namely, a "semi-strong" equilibrium.

Next, we explore the conditions under which pay or play games are Pareto efficient, i.e., when no scenario that is strictly better for at least a single player and no worse for all others exists. We also quantify the gap in global efficiency (sum of players' costs) between an equilibrium and the optimum solution (which does not take into account players' own selfish agendas).

Lastly, we discuss the implications of our results for two special cases of pay or play games: classical models of immunization [1, 2] and of differential pricing [15, 16]. In particular, we show that the game described in [1] always admits a Pareto efficient pure Nash equilibrium.

Related Work Decision between actions with certain and uncertain outcomes is the subject of much research in decision theory. Indeed, the rich literature about the (so called) value of information, which concentrates on measuring the gain one obtains by acquiring information. See, e.g., [6, 12, 5, 13].

Equilibrium analysis is fundamental to game theory and has recently also received much attention from a computer science perspective. In particular, establishing when different kinds of equilibria (pure Nash equilibrium, strong Nash equilibrium, and more) are guaranteed to exist, and the complexity of computing such equilibria, are two important, and extensively studied, research topics. See, e.g., classical game-theoretic results on the existence of pure Nash equilibria in congestion games [14], potential games [11], and player-specific congestion games [10], and also more recent results on computing equilibria in these environments [3, 4].

We have already mentioned that pay or play games generalizes classical models of immunization and differential pricing. An additional class of games generalized by the pay or play class are Interdependent Security Games [7, 8]. Similarly to immunization games in these games players decide whether to invest in security or not and their decision affects both their own vulnerability and their peers vulnerability.

2 Model and Preliminaries

In pay or play games we have a set of N self-interested players ($|N| = n$), each with two strategies: *pay* or *play.* We denote the choices of the players by a strategy vector $x = (x_1, \ldots, x_n)$. When referring to pure (deterministic) strategy profiles, that is, the scenario that each player selects either pay or play with probability 1, we shall use $x_i = 0$ to indicate that player i chooses the play strategy and $x_i = 1$ to indicate that player i chooses the pay strategy. We denote by $A(x)$ the set of players who choose the play strategy in pure strategy vector x. The cost of player i in pure strategy-vector x, $c_i(x)$, is some fixed number h_i if i pays in x (i.e., $x_i = 1$) and a function of the set of players who play in x, $g_i(A(x))$, if i plays in x. Formally, we define:

$$c_i(x) = \begin{cases} h_i & x_i = 1 \\ g_i(A(x)) & x_i = 0 \end{cases}$$

In cases that all the players have the same cost function we will refer to the fixed cost as simply h and the cost of the play strategy as $g(\cdot)$.

We require $g_i(\cdot)$ to be monotone nondecreasing (that is, as more players choose play the cost of player i should increase). Formally, if $S \subseteq T$ and $i \in S$ then $g_i(S) \leq g_i(T)$.

Recall that a player plays a mixed strategy when he selects some probability distribution over the two actions. For mixed strategies, x_i will denote the probability that player i chooses the pay strategy. (Observe that a pure strategy is a special case of a mixed strategy.) The cost of player i in a mixed strategy vector x, $c_i(x)$, is his expected cost over the induced distribution over pure strategy vectors: $c_i(x) =$

$$x_i \cdot h_i + (1 - x_i) \cdot \textstyle\sum_{S \subseteq N - \{i\}} \prod_{j \in S} (1 - x_j) g_i(S \cup \{i\}).$$

Our focus in this paper is on the Nash equilibria of play or play games that are defined as follows:

Definition 2.1 *A vector of mixed (pure) strategies x is a* mixed *(pure) Nash equilibrium if for every player i and every mixed (pure) strategy x'_i: $c_i(x'_i, x_{-i}) \geq c_i(x)$.*

As common is game theory literature, x_{-i} is used as shorthand for the strategy vector describing all players' strategies but that of player i, and (x_i, x_{-i}) denotes the strategy vector in which player i's strategy is x_i and the other players' strategies are as in x_{-i}.

3 Pure Nash Equilibria

We begin by addressing the natural question of whether a pure Nash equilibrium always exists in pay or play games. We provide an affirmative answer to this question for some subclasses of pay or play games, but show that, in general, the class includes games that do not admit a pure Nash equilibrium. Furthermore, we show that determining whether a specific pay or play game admits a pure Nash equilibrium is hard both from a computational perspective (NP-hardness) and from an information-theoretic perspective (can involve the communication of exponentially many bits). As each player has a two strategies, though not all pay or play games possess a pure Nash equilibrium, all games do admit at least a single mixed Nash equilibrium. We discuss the properties of such equilibria later on.

3.1 Sufficient Conditions for Existence

Note that pay or play games in which (i) the cost functions of all players depend only on the number of players who choose the play strategy *and* (ii) all players have the same cost function belong to the classic game-theoretic category of "congestion games" [14], and so are guaranteed to possess a pure Nash equilibrium. We now show that a sufficient condition for a play or pay game to admit a pure Nash equilibrium is for just one of these two properties to hold.

First, consider pay or play games in which the cost function of the play strategy ($g_i(\cdot)$) of all the players depends only on the number of players who choose the play strategy (and not on their identities). It is not hard to observe that such games belong to the class of *player-specific congestion games.* This class of games was defined by Milchtaich [10], who showed that these games always admit a pure Nash equilibrium. Thus, the following claim holds for pay or play games:

Claim 3.1 *If for every player i there exists a function w_i such that for every $S \subseteq N - \{i\}$,$g_i(S \cup \{i\}) = w_i(|S|+1)$, then a pure Nash equilibrium exists.*

We now show that if the players are symmetric (i.e., all have the same cost function), then a pure Nash equilibrium always exists. We point out that the cost function of the players is allowed to depend on the identities of players who choose to play (and not just on their number).

Claim 3.2 *If all players in a pay or play game are symmetric, then a pure Nash equilibrium of the game always exists and can be computed efficiently.*

Proof: We present a simple greedy algorithm for computing a pure Nash equilibrium in polynomial time: begin with the strategy vector $x = 1^n$ in which all players choose the pay strategy. While there exists a player $i \notin A(x)$ such that $g(A(x) \cup \{i\}) < h$ set $x_i = 0$.

We claim that the resulting strategy vector is a pure Nash equilibrium. Observe that once the algorithm halts every player $i \in A(x)$ has a cost smaller than h, and so prefers the play strategy. On the other hand, every player $j \notin A(x)$ would have a cost greater than h for choosing the play strategy. □

3.2 Computational Hardness

Next, we show that if the costs are both player-specific and can depend on the identities of the players, a pure Nash equilibrium might not exist at all. This is true even when the cost functions are restricted to be submodular [1].

Claim 3.3 *The pay or play class contains games that do not admit a pure Nash equilibrium, even for submodular cost functions.*

Proof: Consider the following game consisting of three players numbered $0, 1, 2$. The cost of player i is defined as: $h_i = 1.5$, $g_i(\{i-1, i, i+1\}) = 2$, $g_i(\{i-1, i\}) = 2$, $g_i(\{i, i+1\}) = 1$, $g_i(\{i\}) = 1$. Where $i+1$ and $i-1$ are computed modulo 3. We show that this game does not admit any pure Nash equilibrium by doing a case by case analysis of all the possible strategy vectors:

- There is no pure Nash equilibrium in which all players choose the play strategy – one of the players can benefit from choosing the pay strategy.
- There is no pure Nash equilibrium in which two players choose the play strategy – if players j and $j+1$ choose the play strategy then the cost of player $j+1$ is 2 and hence he prefers to choose the pay strategy.
- There is no pure Nash equilibrium in which at most a single player chooses the play strategy – if players j and $j+1$ choose the pay strategy then player j can reduce his cost to 1 by switching to the play strategy.

□

We are now ready to show that determining whether a pure Nash equilibrium exists or not is NP-hard. The proof is based on a reduction from a 3-SAT formula to a pay or play game and uses the construction from the previous claim as a gadget.

[1] A cost function $g(\cdot)$ is submodular if for every two sets of players $S \subseteq T$ and for ever player $j \notin T$ it holds that: $g(T \cup \{j\}) - g(T) \leq g(S \cup \{j\}) - g(S)$.

Theorem 3.4 *Determining whether a pure Nash equilibrium exists or not in a pay or play game is NP-hard.*

Proof: Given an instance of 3-SAT we construct the following pay or play instance where all players have the same fixed cost of 1.5 but different cost functions for the play strategy.

- For each variable v_i of the 3-SAT formula, we create two players – t_i and f_i. We construct their cost functions such that whenever f_i chooses to play then t_i prefers to pay and vice versa. Formally, we define for all subsets S such that $f_i \in S$: $g_{t_i}(S) = 2$ and for all S such that $f_i \notin S$: $g_{t_i}(S) = 1$. Similarly, we define for all S such that $t_i \in S$: $g_{f_i}(S) = 2$ and for all S such that $t_i \notin S$: $g_{f_i}(S) = 1$.
- For every clause i we create three players, $a_{3i}, a_{3i+1}, a_{3i+2}$. We find it easiest to define their costs by an example: consider, for instance, $i = (v_j \vee \bar{v}_k \vee v_l)$, if $t_j \notin S$ or $f_k \notin S$ or $t_l \notin S$ then $g_{a_{3i+r}}(S) = 1$ for $r \in \{0, 1, 2\}$. Else, for a set S such that $t_j, f_k, t_l \in S$ and $a_{3i}, a_{3i+1}, a_{3i+2} \notin S$, we reconstruct the example from Claim 3.3 and define:
  - $g_{3i+r}(\{a_{3i+r-1}, a_{3i+r}, a_{3i+r+1}\} \cup S) = 2$
  - $g_{3i+r}(\{a_{3i+r-1}, a_{3i+r}\} \cup S) = 2$
  - $g_{3i+r}(\{a_{3i+r}, a_{3i+r+1}\} \cup S) = 1$
  - $g_{3i+r}(\{a_{3i+r}\} \cup S) = 1$

  where $r + 1$ and $r - 1$ are computed modulo 3.

Claim 3.5 *The 3-SAT formula can be satisfied if and only if the previously defined game admits a pure Nash equilibrium.*

Proof: First assume that the formula is satisfiable. Let ϕ be an assignment satisfying it. We show that the following strategy vector is an equilibrium: every player of type a_i uses the play strategy, player t_i chooses the play strategy if and only if $\phi_i = T$ and player f_i chooses the play strategy if and only if $\phi_i = F$. To verify that this is indeed a Nash equilibrium observe the following: first, for every i player a_i has a cost of 1 which is smaller than the cost of 1.5 for choosing the pay strategy. If player t_i uses the pay strategy, then player f_i does not use the pay strategy – thus the cost of player t_i for using the pay strategy is 1.5, if it instead chooses the play strategy it would pay 2. Player f_i cost is 1 for playing so this is its best response as well. Similarly, one can show that this is also an equilibrium for players t_i and f_i such that t_i uses the play strategy and player f_i uses the pay strategy.

Next, we show that if there exists a pure Nash equilibrium then the formula is satisfiable. Let x be the Nash equilibrium. Clearly it has to be the case that for all pairs f_i, t_i exactly one of the players chooses pay and the other chooses play. Consider the assignment $\phi_i = T$ if $x_{t_i} = 1$ and $\phi_i = F$ if $x_{t_i} = 0$. Assume towards a contradiction that there exists some clause i which is not satisfied by the assignment ϕ. Suppose, for instance, that $i = (v_j \vee \bar{v}_k \vee v_l)$. This implies that, t_j, f_k and t_l all use the play strategy. Therefore, by construction the three players $a_{3i}, a_{3i+1}, a_{3i+2}$ are in the exact same configuration as the nodes in Claim 3.3 and thus a Nash equilibrium does not exist. □ □

3.3 Communication Hardness

We now prove that determining whether a pure Nash equilibrium exists in a pay or play game is also hard from an information-theoretic perspective. Specifically, we consider the problem of determining whether a Nash equilibrium exists in Yao's classic communication complexity model [17]: Suppose that each of the n players in a pay or play game knows only his own cost function and the different players wish to find out whether, when put together, their cost functions induce a game that admits a pure Nash equilibrium. No computational restrictions whatsoever are imposed on the players. We set an exponential (in the number of players, n) lower bound on the number of bits the players must exchange to learn the answer to this question. (Observe that a player cannot always simply reveal his entire cost function to others as its specification can, in general, be exponential in n.)

Theorem 3.6 *Determining whether a Nash equilibrium exists in a pay or play game requires communicating an exponential (in n) number of bits.*

Proof: To prove the lower bound we present a reduction from the well-studied DISJOINTNESS problem from communication complexity theory. In this classical setting, there are two parties 1 and 2, each holding a subset $A_i \subseteq \{1, \ldots, r\}$. The objective in DISJOINTNESS is to distinguish between the following two possibilities: (1) $A_1 \cap A_2 \neq \emptyset$ (2) $A_1 \cap A_2 = \emptyset$.

Classical results in communication complexity establish that solving DISJOINTNESS necessitates (in the worst case) transmitting $\Omega(r)$ bits. For more information the interested reader is referred to [9].

We now show how to construct an n-player pay or play game G such that a pure Nash equilibrium in G exists if and only if $A_1 \cap A_2 \neq \emptyset$ in the DISJOINTNESS instance. Suppose that $r = \binom{\frac{n-6}{2}}{\frac{n-6}{4}}$ (w.l.o.g., let $n = 4k+6$ for some integer $k > 0$). We identify each element $j \in \{1, \ldots, r\}$ with a unique set $S_j \subseteq \{1, ..., \frac{n-6}{2}\}$ of size $\frac{n-6}{4}$. We create $n - 6$ players as follows. For every element $j \in \{1, ..., \frac{n-6}{2}\}$ we create two players v_j and

u_j. We construct their cost functions such that whenever v_j chooses to play u_j prefers to pay and vice versa. Formally, v_j's cost when choosing the pay strategy is 1.5, as for the play strategy, for all subsets of players S such that $u_j \in S$: $g_{v_j}(S) = 2$, and for all S such that $u_j \notin S$: $g_{v_j}(S) = 1$. The cost function of player u_j is defined similarly.

We create 6 more players: t_0, t_1, t_2, and w_0, w_1, w_2. The cost functions of each of the three players t_0, t_1, and t_2 are similar to those in the example from Claim 3.3 and are defined as follows: the cost of player t_i, $h_{t_i} = 1.5$; for any set $S \subseteq \{1, ..., \frac{n-6}{2}\}$ let $V_S = \bigcup_{i \in S}\{v_i\}$; if there is some $j \in A_1$ such that $S_j \subseteq S$, $g_{t_i}(V_S) = 2$; if S_j is not contained in S for any $j \in A_1$, $g_{t_i}(t_{i-1}, t_i, t_{i+1}, V_S) = 2$, $g_{t_i}(t_{i-1}, t_i, V_S) = 2$, $g_{t_i}(t_i, t_{i+1}, V_S) = 1$, $g_{t_i}(t_i, V_S) = 1$, where $i+1$ and $i-1$ are computed modulo 3. The cost functions of each of the three players w_0, w_1, w_2 are defined similarly: the cost of player w_i, $h_{w_i} = 1.5$; for any set $S \subseteq \{1, ..., \frac{n-6}{2}\}$ let $U_S = \bigcup_{i \in S}\{u_i\}$; if there is some $j \in A_2$ such that $S_j^C \subseteq S$, where S_j^C denotes the complement of S_j, then $g_{w_i}(U_S) = 2$; otherwise, $g_{w_i}(w_{i-1}, w_i, w_{i+1}, U_S) = 2$, $g_{w_i}(w_{i-1}, w_i, U_S) = 2$, $g_{w_i}(w_i, w_{i+1}, U_S) = 1$, $g_{w_i}(w_i, U_S) = 1$. $i+1$ and $i-1$ are again computed modulo 3.

Claim 3.7 *There is a Nash equilibrium in the pay or play game G if and only if $A_1 \cap A_2 \neq \emptyset$ in the DISJOINTNESS instance.*

Proof: First consider the scenario that $A_1 \cap A_2 \neq \emptyset$ in the original DISJOINTNESS instance. We show that in this case there is indeed a pure Nash equilibrium in G. Let $j \in A_1 \cap A_2$. For every $i \in S_j$ set the strategy of player v_i to be play and the strategy of player u_i to be pay. For every $i \in \{1, ..., \frac{n-6}{2}\} \setminus S_j$ set the strategy of player v_i to be pay and the strategy of player u_i to be play. Observe that none of the v_i's or u_i's wish to unilaterally deviate from this (still partial) specification of players' strategies as each of these players' strategies is the exact opposite of that of his counterpart. Now, set the strategies of all t_i's and w_i's to be pay. Observe the t_i's do not wish to deviate as the set of v_i-players who chose to play corresponds to the set S_j. Observe also that the w_i's do not wish to deviate as the set of u_i's who chose to play corresponds to the set S_j^C.

Next, we show that if there exists a Nash equilibrium then $A_1 \cap A_2 \neq \emptyset$. We make the following crucial observation: in any Nash equilibrium exactly $\frac{n-6}{4}$ of the v_i's are using the play strategy. To see this, consider a specific Nash equilibrium. Observe that if more than $\frac{n-6}{4}$ v_i's choose to play then in any pure Nash equilibrium their u_i counterparts would choose to pay. This means that less than $\frac{n-6}{4}$ u_i's pay, which in turn means that, by construction, the three players w_0, w_1, w_2 are in the exact same configuration as the nodes in Claim 3.3—this leads to a contradiction, since for the three nodes in this configuration a pure Nash equilibrium does not exist. A similar argument establishes that no less than $\frac{n-6}{4}$ of the v_i's must play in any Nash equilibrium as otherwise the t_i's will find themselves in the same predicament. Consider now the case that exactly $\frac{n-6}{4}$ v_i's play. Observe that the t_i's avoid being in the configuration in Claim 3.3 only if the set of v_i's who play corresponds to some S_j where $j \in A_1$ and the same holds for the w_i players only if the set of u_i who chose play corresponds to S_j^C and $j \in A_2$. Hence, $j \in A_1 \cap A_2$. □ □

4 Strong and Semi-Strong Equilibria

One of the criticism often raised against Nash equilibria is that they are not resilient to deviations by coalitions of players. Hence, games that admit an equilibrium that is resilient against deviations by coalitions are of special interest. Such equilibria are called *strong* equilibria.

Definition 4.1 *An equilibrium x is* strong *if there is no strategy vector y, such that, for every player $i \in \{j | x_j \neq y_j\}, c_i(y) < c_i(x)$. When y is restricted to be a pure strategy vector we say that x is strong with respect to pure deviations.*

We show that pay or play games that admit a pure Nash equilibrium also admit a strong pure Nash equilibrium:

Theorem 4.2 *If there exists a pure Nash equilibrium in a pay or play game then this equilibrium is strong with respect to pure deviations.*

Proof: Let x be a pure Nash equilibrium. Assume towards contradiction that there exists a deviation of the set of players S that reduces the cost of all of them. Observe that S cannot include any player i that previously used the play strategy ($x_i = 0$). The cost of such players is at most h_i since x is an equilibrium and by switching to the pay strategy their cost would be exactly h_i. Thus, the set consists of players that use the pay strategy in x ($x_i = 1$) and deviate to the play strategy. However, by monotonicity, if player i prefers the play strategy when more players are choosing it, then he should also prefer it when a smaller subset is playing it – in contradiction to the fact that x is an equilibrium. □

One might also require the stronger property that an equilibrium would be also resilient against (uncoordinated) mixed deviations. Unfortunately, as the following example demonstrates, Nash equilibria (both pure and mixed) in our games are not necessarily strong with respect to mixed deviations.

Example 4.3 *Consider the following symmetric* 2*-player instance: the cost of the pay strategy is* $2+\epsilon$*, for some small* ϵ*. The cost of the play strategy is* 2 *if both players choose it and* 1 *if only one of them chooses it. The* unique *equilibrium is for both players to choose the play strategy. Observe that this equilibrium is not resilient against mixed deviations: if the two players choose the play strategy each one exhibits a cost of* 2*. On the other hand, if they both deviate and use the mixed strategy of choosing to pay with probability* 1/2 *to play with probability 1/2, their cost is reduced to* $\frac{1}{2}(2+\epsilon)+\frac{1}{2}(\frac{1}{2}\cdot 1+\frac{1}{2}\cdot 2)=\frac{7}{4}+\frac{1}{2}\epsilon$.

On the bright side, as we shall show below, the equilibria of games in our class are resilient against mixed deviations in a slightly weaker sense, called *semi-strong* Nash equilibrium. Roughly speaking, even though players can benefit from a joint deviation, this deviation is not "stable", as there always exists a player who can improve his cost by deviating again. For instance, the players in Example 4.3 could profit from jointly deviating to the mixed strategy $x_i=\frac{1}{2}$. However, after this deviation, each one of the players can decrease his cost even more by deviating to the strategy $x_i=0$. The fact that deviations are not stable renders coalition formation hard (as there will always be a player who has an incentive to "betray" the others and deviate from the plan).

Definition 4.4 *A mixed equilibrium* x *is* semi-strong *if for every mixed strategy vector* y *at least one of the following properties hold:*

1. *There exists a player* i *such that* $x_i \neq y_i$ *and* $c_i(y) > c_i(x)$*.*
2. *There exists a player* i *such that* $x_i \neq y_i$ *and a strategy* $z_i \neq y_i$ *such that* $c_i(z_i, y_{-i}) < c_i(y)$*.*

We are now ready to prove our main positive result: every equilibrium of a pay or play game is semi-strong. The proof is based on the following simple, yet powerful, fact: if player i plays a mixed strategy then his expected cost is exactly h_i, since in a mixed equilibrium the player's two strategies should give the same payoff.

Theorem 4.5 *Every mixed Nash equilibrium in a pay or play game is semi-strong.*

Proof: Consider an equilibrium x, assume towards a contradiction that it is not a semi-strong equilibrium. Let strategy vector y be the one for which the two properties of the definition do not hold. Observe that the second property implies that y is an equilibrium with respect to the players in the set $S=\{i|x_i \neq y_i\}$. This implies that the cost of any player $i \in S$ for which $y_i > 0$ is h_i since he either plays a mixed strategy in an equilibrium or he plays the pure pay strategy. As the maximal cost a player can exhibit in an equilibrium is h_i, this implies that the only players in S are ones for which $y_i = 0$. Now, by monotonicity of the play function, for every player $i \in S$ we have that $c_i(0, x_{-i}) \leq c_i(y) < c_i(x)$, in contradiction to the fact that x is an equilibrium. □

Corollary 4.6 *Every instance of the pay or play class admits at least a single semi-strong Nash equilibrium.*

This quite remarkable property that a semi-strong Nash equilibrium always exists ceases to hold once we remove the restriction that one of the strategies should have a fixed payoff. This is illustrated by the next example which is a variation on the prisoner's dilemma. For ease of exposition, the game is defined in terms of positive utility the players wish to maximize, instead of cost.

Example 4.7 *Consider the following* 3*-player game. Players* 1 *and* 2 *are paired together such that unless they pick the same strategy all the players have a utility of* 0*. When players* 1 *and* 2 *choose the same strategy, the players utilities are defined by the following matrix where players* 1 *and* 2 *are the row player and player* 3 *is the column player.*

	c	d
c	$4,4$	$0,0$
d	$6,0$	$1,1$

For brevity we only show that there is no mixed semi-strong Nash equilibrium. Let p_1, p_2, p_3 *be the cooperation probabilities (strategy* c*) of the three players respectively. Then, player* 1 *uses a mixed strategy if* $4p_2 \cdot p_3 = (1-p_2)(6p_3 + (1-p_3))$*. Similarly, player* 2 *uses a mixed strategy if* $4p_1 \cdot p_3 = (1-p_1)(6p_3+(1-p_3))$*.*

Therefore, we have that players 1 *and* 2 *always play the same strategy, implying* $p_1 = p_2$*. Hence, player* 3 *plays a mixed strategy if:* $4p_1^2 = (1-p_1)^2$*.*

By solving this system of equations we get that: $p_1 = p_2 = 1/3$ *and* $p_3 = 7/9$*. To complete the proof, observe that this is not a semi-strong equilibrium since players* 1 *and* 2 *can deviate to the pure strategy* d *and increase their utility from* $4/3 \cdot 7/9$ *to* $42/9$*.*

5 Pareto Efficient Equilibria

One of the desirable properties of an equilibrium, increasing its stability, is Pareto efficiency. Roughly speaking, a strategy vector is Pareto efficient if any deviation that reduces the cost of one player (or more) strictly increases the cost of at least a single player. More formally:

Definition 5.1 *An equilibrium x is* Pareto efficient *if there is no strategy vector y, such that, for every player $i, c_i(y) \leq c_i(x)$, and for at least a single player the inequality is strict. If y is restricted to be a pure strategy vector we say that x is Pareto efficient with respect to pure deviations.*

We show that any Nash equilibrium of a "generic" pay or play game, i.e., a game in which players' best-responses are unique, is Pareto efficient. Formally, we define generic pay or play games as follows:

Definition 5.2 *A pay or play game is* generic *if for every player i and set of players S such that $i \in S$: $h_i \neq g_i(S)$.*

We now prove the following:

Theorem 5.3 *In a generic pay or play game, any pure Nash equilibrium is Pareto efficient with respect to pure deviations.*

Proof: Consider a Nash equilibrium x. Assume towards a contradiction that x is not Pareto efficient. Let y be a deviation reducing the cost of at least a single player. Define $S = \{i | x_i \neq y_i\}$. By the assumption that this is a generic game, we have that the cost of every player i choosing the play strategy in x is strictly less than h_i. Therefore, it has to be the case that for all players $j \in S$ it holds that $x_j = 1$. Now, similarly to our argument for the strong Nash equilibrium in Theorem 4.2, if there is a set of players that can reduce their cost by jointly switching from the pay strategy to the play strategy, then by monotonicity it is beneficial for a single player to perform this deviation. This is in contradiction to the fact that x is a Nash equilibrium. □

Corollary 5.4 *In a pay or play game, any pure Nash equilibrium in which every player i who uses the play strategy incurs a cost strictly lower than h_i is Pareto efficient with respect to pure deviations.*

Unfortunately, the previous theorem no longer holds for mixed deviations, as Example 4.3 illustrates.

Next, we demonstrate the importance of requiring the game to be generic. By tweaking the example from Claim 3.3 we create an instance in which in every equilibrium some players are indifferent between the two strategies, but their choice effects other players' cost.

Claim 5.5 *The class of pay or play games contains games that possess pure Nash equilibria, and all such equilibria are not Pareto efficient.*

Proof: Consider the following game which includes four players numbered $0, 1, 2, 3$. The cost of player $i \in \{0, 1, 2\}$ is defined as: $h_i = 1.5$ for the pay strategy. $g_i(\{i-1, i, i+1\}) = 2$, $g_i(\{i-1, i\}) = 2$, $g_i(\{i, i+1\}) = 1.5$, $g_i(\{i\}) = 1$. Where $i+1$ and $i-1$ are computed modulo 3. The cost of player 3 is: $h_3 = 10$ and $g_3(S) = |S|$ for a set S such that $3 \in S$. Observe that in all Nash equilibria exactly one player of the players $0, 1, 2$ chooses the pay strategy and the rest of the players choose the pay strategy. First, without loss of generality, we show that the strategy vector in which player 0 is the only one using the pay strategy is an equilibrium. Notice that player 1 is indifferent between the two strategies as both have a cost of 1.5. Players 2 and 3 strictly prefer the play strategy, hence this is an equilibrium. Next, we do a case by case analysis and show that any strategy vector in which the number of players using the pay strategy is not exactly one, is not an equilibrium.

1. There is no pure Nash equilibrium in which none of the players choose the pay strategy, since in this case one of the players $\{0, 1, 2\}$ can reduce its cost by choosing the pay strategy.
2. There is no pure Nash equilibrium in which two players (or more) choose the pay strategy. Clearly player 3 never choose the pay strategy. Now, if players j and $j+1$ choose the *pay* strategy then if player j switches to the pay strategy it reduces its cost to 1.

Observe that this equilibrium, in which a single player $i \in \{0, 1, 2\}$ chooses the pay strategy is not Pareto efficient. The reason is that, if player $i+1$ switches to the pay strategy then player 3 strictly benefit and the cost of the rest of the players remains the same. □

In the next section, we present in more depth one of the well studied games that belong to the pay or play class and show that every instance of this game admits a Pareto efficient pure Nash equilibrium.

6 Examples: Vaccination Games and Differential Pricing

The pay or play class is quite broad. In this section we focus on two well-studied subclasses of games that is contained in this class: vaccination games and differential pricing.

6.1 Vaccination Games

We first discuss the class of games presented by Aspnes et al. [1], which we refer to as "vaccination games". A vaccination game is played on a network G with $|V| = n$ nodes that are the players of the game. Each player is faced with the following decision: buy a vaccination or not. If a player buys a vaccination then he pays a fixed cost, denoted by c. Else, the player risks getting his computer infected and exhibiting a loss of l. After all the players make their decisions one of the nodes in the network is selected uniformly at random to be infected by some virus. Next, the virus spreads in discrete rounds, such that in every round all the neighbors of every infected node that are not vaccinated get infected.

More formally, let x be the strategy vector describing the decisions of the players whether to get the vaccine or not. $x_i = 1$ for a player that chooses to get the vaccine (pay) and $x_i = 0$ for a player that chooses not to get it (play). Denote by $R(x)$ the set of nodes choosing the pay strategy – getting the vaccine. Let G_x be the attack graph that is constructed by removing all nodes in $R(x)$ and all their incident edges. The cost of the play strategy for node i depends on the size of the connected component in G_x that i belongs to and the loss l. More precisely, the expected cost of the play strategy for a node i in a connected component of size k_i in G_x is $\frac{k_i}{n} \cdot l$. It is not hard to see that this function is monotone increasing in the number of players choosing the play strategy and thus, this game belongs to the the pay or play framework.

It is shown in [1] that a pure Nash equilibrium for this game always exists. The proof is via a potential function, which relates the players' best responses to the size of the connected components in the attack graph. Let $\alpha = \frac{cn}{l}$. The set of pure Nash equilibria is characterized in [1] as follows: (1) every connected component of G_x has a size of at most α; and (2) for every player $i \in R(x)$ the size of its connected component in G_x when node i is added to the graph together with all its incident edges is at least α.

By utilizing the framework of pay or play games, we are able to prove a new result for vaccination games – showing that a pareto-optimal Nash equilibrium always exists. As was discussed in the previous section, this property does not hold for pay or play games in general.

Theorem 6.1 *The vaccination game admits a Pareto efficient Nash equilibrium.*

Proof: Assume without loss of generality that $l = 1$. This implies that the cost of the play strategy for player i in strategy vector x is simply the size of its connected component in $G_x + i$ divided by n. We refer to this as its infection probability. We show that there exists an equilibrium in which the infection probability of every node choosing the play strategy is strictly less than c. In other words, this implies that the size of every connected component of G_x is strictly smaller than c. By Corollary 5.4 we have that this implies the equilibrium is pareto-optimal which completes the proof.

Assume towards a contradiction that in every equilibrium x there exists a connected component of G_x of size c. Let x be an equilibrium for which G_x has the minimal number of connected components of size c. Note that in case one of the connected components is not a tree, then it is possible to construct a new equilibrium with less connected components of size c. If the connected component is not a tree then there exists a node that can change its strategy to pay without harming the connectivity of its connected component in the attack graph. Denote this player by i. The new strategy vector is an equilibrium since player i is indifferent between the two strategies. The only other affected players are ones in i's connected component that still want to use the play strategy and ones using the pay strategy which are adjacent to i's connected component. The adjacent nodes do not want to change their strategy to play since by doing that they will be a part of a connected component of size at least c, thus they do not want to switch.

Thus, it remains to handle the case in which all connected components of size c are trees. Consider a leaf i in one such tree, if this leaf is not connected to any other node (except its parent in the tree), then it can switch its strategy to play and it is still an equilibrium. Otherwise, it is connected to nodes who choose the pay strategy, denote this set of nodes by S. Go over the nodes in S in some arbitrary order, for each node j check the size of its connected component, if it is at most $c - 2$ change its strategy to play and continue. We claim that the resulting strategy vector is an equilibrium with a smaller number of connected components of size c. Observe that by construction the size of each connected component of the attack graph of the new strategy vector including neighbors of i, is smaller than c, therefore all nodes using the play strategy prefer it over the pay strategy. Also by construction, all the nodes in S that use the pay strategy would be in a connected component of size at least c if they decide to switch their strategy. Thus, the new strategy vector y is an equilibrium such that G_y has less connected connected components of size c than G_x, a contradiction. □

6.2 Differential Pricing

Lastly, we briefly discuss another well-studied environment: differential pricing. Consider the following scenario: n buyers are interested in purchasing some good, say a laptop. Each buyer has two options: (1) he can buy a laptop for a fixed price p (there is a large enough supply of laptops to sell to all buyers); (2) take part in a lottery in which $k < n$ laptops will be assigned to k bidders, uniformly at random, and each buyer who receives a laptop is charged a lower price $q < p$. (Of course, if there are less than k buyers who decide to participate in the lottery, each of these buyers will be given a laptop). Observe that this can easily be formulated as a pay or play game. We note that every such environment admits a pure Nash equilibrium (and it is, in fact, a congestion game).

7 Price of Anarchy and Price of Stability

A natural metric for measuring the efficiency of a pure Nash equilibrium is by comparing its social cost (the sum of all players' costs) and the cost of the socially optimal solution (the strategy vector minimizing the sum of all players' costs). We present several simple results bounding the ratio between the optimal solution and worst pure Nash equilibrium (a.k.a price of anarchy) and the ratio between the optimal solution and best pure Nash equilibrium (a.k.a "price of stability") with respect to different restrictions on the cost functions. We begin with a positive result showing that for a very restricted subclass of pay or play games the price of anarchy is 2:

Claim 7.1 *If all players have the same submodular cost functions, and the cost function does not depend on players' identities, then the (pure) price of anarchy is bounded by* 2.

Proof: Consider a specific pure Nash equilibrium x and optimal solution o. Denote by k_x and k_o the number of players using the play strategy in x and o receptively. Observe that if $k_o \leq \frac{n}{2}$, then at least $n/2$ players choose to pay and hence the cost of the optimal solution is at least $\frac{n}{2} \cdot h$. The cost of the Nash equilibrium is at most $n \cdot h$, since players can always choose the pay strategy and pay h. Thus, the price of anarchy for this case is at most 2.

We are left with the case that $k_o > \frac{n}{2}$. Observe that this trivially implies that $k_x \leq 2k_o$. Also, it is not hard to see that $k_o \leq k_x$. Now, consider the difference in cost between the pure Nash equilibrium and the optimal solution: $c(x) - c(o) = ((n-k_x)h + k_x \cdot g(k_x)) - ((n-k_o)h + k_o \cdot g(k_o)) = (k_o - k_x)h + k_x \cdot g(k_x) - k_o \cdot g(k_o)$. The fact that $k_n < \frac{k_o}{2}$, together with the submodularity of the cost function, and the fact that the cost function is nondecreasing, imply that $g(k_x) \leq g(2k_o) \leq 2g(k_o)$. Hence, $c(x) - c(o) < (k_o - k_x)h + 2k_x \cdot g(k_o) - k_o g(k_o) = (k_o - k_x)h + (2k_x - k_o)g(k_o) \leq k_x \cdot g(k_o) \leq n \cdot g(k_o) \leq c(o)$, where the last two inequalities follow from the simple observation that $h \geq g(k_o)$ □

Next, we show that once we lift either of the two restrictions previously imposed: (1) all players have the same cost functions, (2) the cost function depends only on the number of players choosing the play strategy, the price of stability can be very high:

Claim 7.2 *The (pure) price of stability of a game with player-specific cost functions that are not dependent on players' identities can be linear in n.*

Proof: Consider the following n-player instance, for player 1, $h_1 = n + \epsilon$ and $g_1(S) = |S|$ for $i \in S \subseteq N$. For player $j \neq 1$ $h_j = 2\epsilon$ and $g_j(S) = \epsilon$ for $j \in S \subseteq N$. In the optimal solution of this instance, player 1 is the only one choosing the play strategy – the cost is $1 + 2(n-1)\epsilon$. On the other hand, in the unique Nash equilibrium all players choose the play strategy, the social cost in this case $n + (n-1)\epsilon$. □

Claim 7.3 *If all players have the same submodular cost function (possibly depends on the players' identities) then the (pure) PoS can be linear in n.*

Proof: Consider the following instance where $h = 1 + \epsilon$ and for any set S such that $1 \notin S$ we define $g(\{1\} \cup S) = 1$ and $g(S) = 0$. Then, in the optimal solution player 1 chooses the pay strategy, for a social cost of $1+\epsilon$. In any Nash equilibrium all players choose the play strategy for a total cost of n. □

8 Conclusions

We introduced the pay or play framework, which captures a simple scenario in which decision makers select between certain and uncertain outcomes, and the realization of the uncertain outcome is solely dependent on the decision makers and not on "nature". We studied the properties of equilibria (existence, efficiency, complexity, and more) in pay or play games from both a game-theoretic perspective and a computational perspective. Our main positive result established that games in this class always possess a semi-strong equilibrium. We regard our results for pay or play as a first step, and believe that further exploring the game-theoretic and computational properties of this class of games (and its subclasses) can provide valuable insights into strategic decision making under uncertainty.

References

[1] James Aspnes, Kevin L. Chang, and Aleksandr Yampolskiy. Inoculation strategies for victims of viruses and the sum-of-squares partition problem. *J. Comput. Syst. Sci.*, 72(6):1077–1093, 2006.

[2] Po-An Chen, Mary David, and David Kempe. Better vaccination strategies for better people. In *ACM Conference on Electronic Commerce*, pages 179–188, 2010.

[3] A. Fabrikant, C. H. Papadimitriou, and K. Talwar. The complexity of pure nash equilibria. In *STOC*, pages 604–612, 2004.

[4] S. Hart and Y. Mansour. The communication complexity of uncoupled nash equilibrium procedures. In *STOC*, pages 345–353, 2007.

[5] David Heckerman, Eric Horvitz, and Blackford Middleton. An approximate nonmyopic computation for value of information. In *UAI*, pages 135–141, 1991.

[6] B.A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2:22–26, 1996.

[7] Michael Kearns and Luis E. Ortiz. Algorithms for interdependent security games. In *In Advances in Neural Information Processing Systems*. MIT Press, 2004.

[8] Howard Kunreuther and Geoffrey Heal. Interdependent security. *Journal of Risk and Uncertainty*, 26(2-3):231–249, 2003.

[9] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1996.

[10] Igal Milchtaich. Congestion games with player-specific payoff functions. *Games and Economic Behavior*, 13(1):111–124, 1996.

[11] D. Monderer and L.S. Shapley. Potential games. *Games and Economic Behavior*, 14:124–143, 1996.

[12] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, California, 1988.

[13] Kim-Leng Poh and Eric Horvitz. A graph-theoretic analysis of information value. In *UAI*, pages 427–435, 1996.

[14] R.W. Rosenthal. A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory*, 2:65–67, 1973.

[15] H. Varian. Price discrimination and social welfare. *American Economic Review*, 75(4):870–875, 1985.

[16] H. Varian. Price discrimination. In R. Schmalensee and R. Willig, editors, *Handbook of Industrial Organization: Volume I*, pages 597–654. 1989.

[17] A. C. Yao. Some complexity questions related to distributive computing. In *ACM Symposium on Theory of Computing*, pages 209–213, 1979.

Evaluating computational models of explanation using human judgments

Michael Pacer **Joseph Williams** **Xi Chen** **Tania Lombrozo** **Thomas L. Griffiths**
{mpacer, joseph_williams, c.xi, lombrozo, tom_griffiths}@berkeley.edu
Department of Psychology, University of California at Berkeley, Berkeley, CA, USA

Abstract

We evaluate four computational models of explanation in Bayesian networks by comparing model predictions to human judgments. In two experiments, we present human participants with causal structures for which the models make divergent predictions and either solicit the best explanation for an observed event (Experiment 1) or have participants rate provided explanations for an observed event (Experiment 2). Across two versions of two causal structures and across both experiments, we find that the Causal Explanation Tree and Most Relevant Explanation models provide better fits to human data than either Most Probable Explanation or Explanation Tree models. We identify strengths and shortcomings of these models and what they can reveal about human explanation. We conclude by suggesting the value of pursuing computational and psychological investigations of explanation in parallel.

1 Introduction

Representing statistical dependencies and causal relationships is important for supporting intelligent decision-making and action – be it executed by human or machine. Causal knowledge not only allows predictions about what will happen, but is also used in explanations for events that have already occurred. For example, a set of symptoms might be explained by appeal to a particular disease, or an electrical circuit failure by appeal to a set of faulty gates.

Previous work in machine learning has provided a range of models for what counts as an explanation in cases involving a known causal system and observed effects.These models differ in what they allow as potential explanations or 'hypotheses' as well as in the objective function they aim to maximize (for a review, see Lacave and Díez [2002]). For example, one approach says hypotheses are settings for all unknown variables where you then choose the hypothesis that maximizes a posteriori probability given observed data [Pearl, 1988]; another allows hypotheses to be any nonempty variable setting and selects the hypothesis that maximizes the probability of observations under that hypothesis relative to every other hypothesis [Yuan and Lu, 2007]. While these models differ in their formal properties, arguments for one model over another typically come down to which provides a better fit to researchers' intuitions about the best explanations in a given case.

In this paper we evaluate four formal models of explanation by empirically investigating their fit to human judgments. Our aims are threefold. First, methods from cognitive psychology allow us to test how well competing models correspond to general human intuitions, rather than the intuitions of a small group of researchers. Second, by using human judgment as a constraint on formal models of explanation, we increase the odds of choosing an objective function with interesting properties for learning and inference. A growing literature in psychology and cognitive science suggests that generating and evaluating explanations plays a key role in learning and inference for both children and adults (for a review, see Lombrozo [2012]), so effectively mimicking these effects of explanation in formal systems is a promising step towards closing the gap between human and machine performance on challenging inductive problems. Finally, formal models of explanation that successfully correspond to human judgment can contribute to the psychological study of explanation, as almost no formal models of explanation generation or evaluation have been proposed within the psychological sciences.

We present two experiments in which we gave people information about a causal system and had them either generate explanations (Experiment 1) or evaluate explanations (Experiment 2). The causal systems can be formally defined by Bayesian networks and correspond to those used in prior work to differentiate among models of explanation [Nielsen et al., 2008, Yuan and Lu, 2007]. Across two versions of two causal structures and across both experiments, we find that the Causal Explanation Tree [Nielsen et al., 2008] and Most Relevant Explanation [Yuan and Lu, 2007] models provide better fits to human data than either Most Probable Explanation [Pearl, 1988] or Explanation Tree models [Flores et al., 2005]. The results of our experiments identify strengths and shortcomings of these models, ultimately suggesting that human explanation is poorly characterized by models that emphasize only maximizing posterior probability.

2 Bayesian networks

A Bayesian network provides a compact representation for the joint probability of a set of random variables, $\mathcal{X}$, which explicitly represents various conditional independence statements between variables in $\mathcal{X}$. We specify a directed acyclic graph with a node corresponding to each variable in $\mathcal{X}$. We say that each node $X \in \mathcal{X}$ has a set of "parent nodes" ($\mathrm{Pa}(X)$), and that this gives us conditional probability distributions for every X given its parents $p(X|\mathrm{Pa}(X))$. We assume that the full joint probability distribution can be specified this way, i.e., that $p(\mathcal{X}) = \prod_{X \in \mathcal{X}} p(X|\mathrm{Pa}(X))$. This is equivalent to assuming that X is independent of all nondescendent variables given its parents, and allows us to use the structure of the graph to read off which conditional independence relations must hold between the variables [Pearl, 1988].

Figure 1 shows an example of a Bayesian network specifying conditional probability distributions between random variables. The graph on the left (*Pearl*, named after Pearl [1988]) represents whether a particular alien has a disease (D), whether that alien has a genetic risk factor for that disease (G), and whether or not the alien was vaccinated for the disease (V). The graph on the right (*Circuit*) can be interpreted as a circuit that always receives input and for which we can measure the output. A, B, C, and D are gates that, if functional, break the circuit, stopping the input from reaching the output. Each gate has an independent probability of failing and allowing current to cross through it. If the current can travel from the input to the output via any path made possible by a set of failed gates, then there will be output. These two examples hint at the richness of the Bayesian network formalism. We will continue to refer to these graphs throughout, which are the basis for our stimuli in Experiments 1 and 2, with the parameter values indicated in Figure 1.

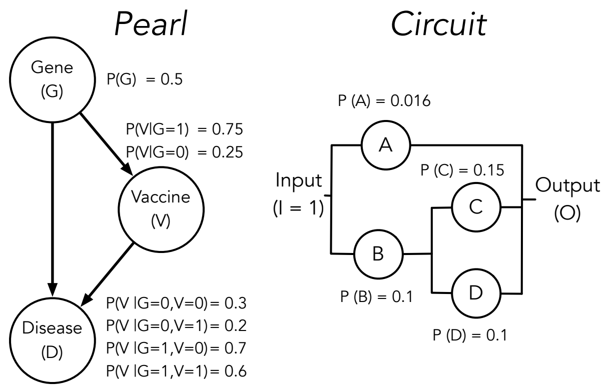


Figure 1: The *Pearl* and *Circuit* networks used in our experiments; as in Pearl [1988], Nielsen et al. [2008], Yuan and Lu [2007] and Yuan et al. [2011].

2.1 Explanations in Bayesian networks

Suppose we observe the values for k of the variables in a graph, $\{O_1 = o_1, \ldots, O_k = o_k\}, \forall i\ O_i \in \mathcal{X}$. We may not wish to explain every observation, so let us call the variables we want to explain O_{exp}, with values o_{exp}. These values o_{exp} are the "target" of our explanation, or the *explanandum*, which is a subset of $\mathcal{O}$, the set of possible observation sets. We will refer to $\hat{O}$ as the set of variables that were observed and $\hat{o}$ as the observed values. Then $O_{\text{not-exp}} = o_{\text{not-exp}}$ are those variables that are observed and unexplained (or $O_{\text{not-exp}} \equiv \hat{O} \setminus O_{\text{exp}}$).

A candidate explanation (the *explanans*, or "hypothesis") is a set of variable assignments for some of the variables not in O_{exp}. We exclude O_{exp} to avoid circularity, though elements in $\hat{O} = \hat{o}$ but not in O_{exp}(i.e., observed but unexplained variables) could be included. However, we should note that most models require that every observed variable be explained; formally $\hat{O} \equiv O_{\text{exp}}$. For the sake of clarity, a hypothesis (our term for potential explanans henceforth) will be represented by h, the variables assigned in that hypothesis by H, and the set of hypotheses (treating each set of assignments as a separate hypothesis) as $\mathcal{H}$.

The first question a formal account of explanation must answer is which variables should be used in constructing $\mathcal{H}$. One possibility is for every explanation to include an assignment for every unobserved variable. However, Bayesian networks often use variables not meant to correspond to real entities in the world (e.g., a noisy-or gate for combining the influence of two causes). Additionally, there are often many variables that are not invoked in an explanation, and so a notion of "relevance" can be useful, allowing assignments to a subset of the unobserved variables (or even variables that are observed but not in O_{exp}).

Some models first generate $\mathcal{H}$ and then evaluate each hypothesis and rank them accordingly. Others "grow" their hypotheses by iteratively adding variables based on their ability to improve the explanation, stopping when the hypothesis cannot be improved further [Flores et al., 2005, Nielsen et al., 2008]. The hypotheses under consideration can then be evaluated and ranked, but note that what counts as an improved hypothesis and what counts as a better explanation can be based on different criteria even within the same model. Some models aim to maximize the probability of the hypothesis given the observations ($p(h|\hat{o})$) [Pearl, 1988, Shimony, 1991]. Some models are more concerned with other metrics, such as the relative likelihood of the observations under one hypothesis ($p(\hat{o}|h)$) compared to the rest of the hypothesis set [Yuan and Lu, 2007, Yuan et al., 2011]. And some models aim to maximize how much information is gained about the explanandum were the hypothesis assumed or made to be true [Flores et al., 2005, Nielsen et al., 2008].

We now introduce the four models that we consider in this paper — Most Probable Explanation [Pearl, 1988], Most Relevant Explanation [Yuan and Lu, 2007], Explanation Trees [Flores et al., 2005], and Causal Explanation Trees [Nielsen et al., 2008].

2.2 Most Probable Explanation (MPE)

Most Probable Explanation (**MPE**) ranks highly hypotheses with the most probable assignments to all unobserved variables, conditioning on $\hat{O}$. That is, every h in $\mathcal{H}$ includes an assignment for every variable in $\mathcal{X} \setminus \hat{O}$.[1] This model leverages the intuition that the best explanation is one that is most probable given what we have observed [Pearl, 1988]. The result is

$$MPE = \arg\max_{h \in \mathcal{H}} p(h|\hat{o}). \tag{1}$$

2.3 Most Relevant Explanation (MRE)

Rather than choosing the hypothesis that maximizes the probability of the unobserved variables given the observed values, we could choose values for the unobserved variables to maximize the probability of the observations ($\arg\max_{h \in \mathcal{H}} p(\hat{o}|h)$). Methods that pursue this route are known as likelihood models.

One problem faced by likelihood models is that multiple hypotheses will sometimes give the same high probabilities to the observed data [Nielsen et al., 2008]. For example, consider the case where we know the structure of a causal system like the circuit in Figure 1 from Yuan and Lu [2007]. Likelihood methods would treat any hypothesis containing a union of A, "B and C", or "B and D" as equally good — the current flows equally well (perfectly), regardless of the particular path it takes. This can make it difficult to choose between these explanations within the likelihood framework.

Rather than maximizing the likelihood per se, we can instead choose the hypothesis, h, that has the highest likelihood *relative* to the summed likelihood of all the other hypotheses in $\mathcal{H}$ except for h:

$$\frac{p(\mathcal{O}|h)}{\sum_{h_j \neq h, h_j \in \mathcal{H}} p(\mathcal{O}|h_j)}. \tag{2}$$

Yuan and colleagues' Most Relevant Explanation (**MRE**) model [Yuan and Lu, 2007, Yuan et al., 2011] proposes that the best explanation maximizes this quantity. This term plays an important role in statistics, known as the Generalized Bayes Factor [Fitelson, 2007], and in psychology, as a measure of how *representative* some data is of a hypothesis [Tenenbaum and Griffiths, 2001, Abbott et al., 2012].

2.4 Tree-based models: ET and CET

The methods we have explored so far presume that you have $\mathcal{H}$ and then evaluate each hypothesis to determine which is best. However, in cases where the variable set is large, this can be difficult and computationally prohibitive. A class of *tree*-based models addresses this problem by using an iterative process for arriving at explanations. These models construct an explanation piece-wise, adding variables to the hypothesis one at a time, by choosing the best variable, assigning the variable a value and repeating until no further gains can be made. The resulting hypotheses are then evaluated based on some criteria, producing a list of explanations ranked by their goodness. Models differ in how they choose the best variable to add, how they decide to stop, and how they then evaluate the resulting hypotheses.

The Explanation Tree (**ET**) model — as proposed by Flores et al. [2005] — determines which variable carries the most information about the rest of the unknown nodes, conditioned on what is already known. In **ET** what is already known includes $\hat{O}$ and any variables included in hypotheses farther up the tree. This means that at the beginning (when the hypothesis is $\emptyset$) the model selects the node that provides the most information about the rest of the unobserved variables conditioned on $\hat{O}$. Formally, we grow h' (the hypothesis up to that point) by choosing the X_i as the maximum of $\sum_Y \mathrm{INF}(X_i; Y|\hat{O}, h')$, where Y is shorthand

[1]We might allow $h \in \mathcal{H}$ to include only those variables that are relevant for explaining $\hat{O}$. This is known instead as the maximum a posteriori model. There are a variety of possible relevance criteria as explored by De Campos et al. [2001], but this problem is substantially more computationally complex than **MPE**. Here, we focus on **MPE**.

for $\mathcal{X} \setminus \{\hat{O} \cup h' \cup \{X_i\}$, or all of the variables not observed, included in the current hypothesis or currently under consideration, and INF$(\cdot)$ is a metric of informativeness. In our calculations we will use *mutual information* as our INF$(\cdot)$, as in Nielsen et al. [2008].[2]

Once a variable is chosen, each assignment creates a new branch, and that assignment is added to the interim hypothesis h', and is effectively treated as an observed variable. The process is then repeated until adding any more variables is deemed to provide a hypothesis with a probability that is too low, as defined by parameter β_{ET}, or to carry too little information, as defined by parameter $\alpha_{\mathbf{ET}}$. This process provides multiple, mutually exclusive explanations that can vary in their complexity based on how much information the complexity buys.[3] Once these hypotheses are assembled, the model ranks the explanations by the posterior probability of each branch of the tree – i.e., how likely each hypothesis is, given the observed data.

Up to this point every model we have considered assumes the set of observed data is the data we are explaining, or $\hat{O} \equiv O_{\text{exp}}$. The **ET** model further assumes that we aim to reduce uncertainty of the entire variable set $\mathcal{X}$ in deciding which variables are ostensibly relevant to our explanandum, O_{exp}. However, these assumptions can be problematic. For example, in **ET**, a variable that is unrelated to O_{exp} but carries a lot of information about other unknown variables may be added to the hypothesis despite its irrelevance to our explanans.

The Causal Explanatory Tree (**CET**) model introduced by Nielsen et al. [2008] addresses these weaknesses. Rather than using traditional measures of information such as mutual information, **CET** uses *causal information flow* [Ay and Polani, 2008] to decide how the tree will grow. Causal information flow uses the post-intervention distribution on nodes (as proposed in Pearl [2000]) rather than considering the joint probability distribution "as is". To extend Ay and Polani [2008]'s analogy, imagine pouring red dye into a flowing river. You could identify which way is downstream by tracking the red streak that results; if you were to pour in the dye just after a fork in the river, you would not find red dye in the other half of the fork. Now consider the case of a static, dammed river — a river that does not flow. If you poured the dye just after the fork, redness would gradually diffuse through the water, eventually reaching the other path from the fork and tinting the whole river. In this case, there is no concept of something being 'downstream'. Causal information attempts to capture the notion of 'downstream' influence that is absent in traditional mutual information.

We denote post-intervention distributions with a " $^-$ " on a conditioned variable. If we have variables W, X, Y, Z, where we have observed $W = w$, intervened on Z (giving us post-intervention values $\bar{Z} = \bar{z}$), then the causal information passed from X to Y is,

$$\sum_{x \in X} p(X = x | W = w, \bar{Z} = \bar{z}) \times \sum_{y \in Y} p(Y = y | \bar{X} = \bar{x}, w, \bar{z}) \log \frac{p(y|\bar{x}, w, \bar{z})}{p(y|w, \bar{z})} \quad . (3)$$

This allows us to specifically ask the degree to which a variable ($X \equiv X_i$) influences the explained data ($Y \equiv O_{\text{exp}}$), treating the non-explained data as observed ($W \equiv O_{\text{not-exp}}$) and previous parts of the explanation as intervened on ($Z \equiv h'$). This solves the problem of distinguishing between explained and unexplained observations ($W \neq Y$). It also allows us to maximize information about the O_{exp} rather than $\mathcal{X} \setminus \hat{O}$ as in **ET**. However, like **ET**, the **CET** model proposes variables iteratively, until no remaining variables add more causal information than the criterion $\alpha_{\mathbf{CET}}$. Then each branch is assigned the score $\log\left(\frac{p(O_{\text{exp}}|\bar{h}', O_{\text{not-exp}})}{p(O_{\text{exp}}|O_{\text{not-exp}})}\right)$ where $\bar{h}'$ is the total set of assigned values in a hypothesis at a branching point.

3 Comparing model and human judgments about explanations

We now compare the prediction of these four models against human judgments when both generating and evaluating explanations. We focus on explanations in the two Bayesian networks shown in Figure 1. The *Pearl* structure is derived and parameterized as in Nielsen et al. [2008]; the *Circuit* graph and its parameters are taken from Yuan and Lu [2007]. These networks have been used previously to distinguish between the performance of different models. Each network consists of several binary variables, prior probabilities on those variables, and relationships between variables. We consider the case where only one variable is observed, in *Pearl* $D = 1$ and in *Circuit* $O = 1$, and these act as both $\hat{O}$ and O_{exp}, i.e., each is the only variable we observe and explain in that structure.

The models diverge in how they rank explanations in *Pearl* and *Circuit*. In past research, the *Pearl* structure was used by Nielsen et al. [2008] to argue in favor of the **CET**, and the *Circuit* structure was used by Yuan and Lu [2007] to argue in favor of the **MRE**.[4]

[2]Flores et al. [2005] consider several versions of INF.

[3]Mutual exclusivity refers to the fact that once a variable is assigned, it holds through the rest of the tree.

[4]The **CET** had not been published by the writing of Yuan and Lu [2007]. Yuan et al. [2011] addresses **CET**

By drawing from distinct research lines we aim to be as fair as possible in testing the models.

In addition to being useful for distinguishing between models, these structures have properties that are particularly interesting from a psychological perspective. The *Pearl* structure includes complex causal dependencies that cannot be easily captured by the paradigms used in cognitive psychology. The *Circuit* structure contains explanations with equal (perfect) likelihoods for the observation, but which vary in the number of variables cited in the explanation. Research on people's preferences for simplicity bear on this case, which shows that people may choose an explanation with fewer causes even if it is less likely than other more complex alternatives [Lombrozo, 2007].

In the past, researchers used the match between their own explanatory intuitions and the models' predictions to provide support for their model. However, this method can be problematic: Nielsen et al. [2008] and Yuan and Lu [2007] conflict in their intuitions, leaving us in a quandary. We generalize the intuition-matching approach using two experiments in which we ask people to generate (Experiment 1) and evaluate (Experiment 2) explanations in cases formally equivalent to *Circuit* and *Pearl*. We used **MPE**, **MRE**, **ET**, and **CET** to rank the quality of various explanations, and analyze these rankings as they compare to the rankings derived from human explanations. By appealing to a wider array of human judgments we hope to extricate ourselves from this quandary.

4 Experiment 1: Generation

4.1 Participants

We recruited 188 participants through Amazon Mechanical Turk; 9.6% of those failed to complete the study, did not consent to taking the study, or did not follow the instructions, and 35.9% failed at least one explicit reading/attention check. This left 109 participants for analysis (M(age) = 27.7, %-Female = 29.3%).

4.2 Materials & procedure

Participants were randomly assigned to either the *Pearl* or *Circuit* structure. They then were assigned to one of two semantically-enriched stories embodying a causal structure, involving either novel alien diseases or the ecology of lakes. For example, one of the two scenarios adapted from the *Circuit* structure taught participants about the effects of novel diseases on producing a kind of fever.

For this scenario, participants received facts about the base rates of four novel diseases (corresponding to $p(A)$, $p(B)$, $p(C)$, and $p(D)$), and information allowing them to understand which diseases would produce the fever, which would only occur in the presence of two proteins X and Y. One disease (corresponding to A) produced both the necessary proteins and thereby caused the fever. The second disease (corresponding to B) produced one of these proteins, and when paired with either the third and/or the fourth diseases (i.e., C or D) which produced the other protein, would be sufficient to cause the fever. X and Y were added to provide an intuitive mechanism outside of the domain of circuits that describes the complexities of *Circuit*'s causal relations. Probabilities were presented as frequencies (out of 1000) and act as realizations of the probabilities in the graphs in Figure 1.

In order to ensure that participants were paying attention, we asked questions that required simply reading the information off a figure (e.g., "Out of 1000, how many aliens have [disease A]?"). Participants who failed any comprehension questions were excluded from subsequent analyses. To ensure that participants' judgments were not limited by memory, the base rates and causal structure were available when answering these reading checks as well as during the generation portion of the experiment. Participants were asked to use the information that had been provided to write down "the **SINGLE BEST EXPLANATION**" for the observed effect (e.g., for a particular alien's fever), where "a 'single' explanation can include more than one causal factor." Participants were explicitly asked not to list multiple possible explanations, but rather to "identify the one explanation that you think is the best." This was meant to exclude what we call "disjunctive" explanations like "It was A or B and C and not D", or, formally, as $A = 1 \cup \{B = 1 \cap C = 1 \cap D = 0\}$.

4.3 Results and discussion

Participants' explanations were coded by an assistant blind to the authors' hypotheses. The coder's goal was to identify which variables were mentioned and what values were assigned to those variables. We excluded participants who gave a response that conflicted with our instructions, such as providing a disjunctive explanation.

In *Circuit*, most participants provided explanations that fell into one of two options: BC (43%) or A (40%), and, in *Pearl*, most participants chose one option: they attributed the disease to the presence of a genetic risk factor and not receiving the vaccine (73%,

but that work involves more complicated scenarios than those considered here.

see Figure 1).

For the explanations participants generated, we computed measures of explanation quality under each of the four models and saw which models gave better scores to those explanations that were generated more frequently. This process provides us a rank for each participant's explanation according to each of the models and a rank of how frequently each explanation was generated, which allows us to calculate a Spearman rank-order correlation between participant's aggregate explanation choices and the models' predictions, see Table 1.

Note, we used two versions of the *tree*-algorithms: one where explanations not reached by the tree received the lowest possible rank (which we give the subscript "$_{\text{tree}}$"), and one where we ignored these exclusions and applied the evaluation criteria used at each branch point. The tree models were designed to both generate and evaluate explanations "on the fly", but it is not clear whether the way models *generate* explanations has led to their success in previous literature. Model success (or failure) may be the result of the branch evaluation criterion, rather than the result of the algorithm for generating hypotheses. This is why we analyze these parts of the algorithms separately.

We find that **MRE** and **CET** are most consistent with participants' judgments (though they still only reach marginal significance in the *Circuit* case). In contrast, for both structures, models that rely only on an assignment's probability (i.e., **MPE** and **ET**) poorly predict the explanations that people generate (in *Circuit*, **MPE** had a negative coefficient).

The major weakness of the tree versions of **CET** and **ET** lies in the fact that once a node is chosen for expansion, it remains expanded. Thus, mutually exclusive explanations cannot be reached in the same tree. That is, in *Circuit*, A and BC were the two most popular explanations and $A \cap BC = \emptyset$, so the first step to include either A or B will preclude the other explanation. Empirically, participants are roughly split between these two explanations, which suggests that any method that generates a unique best explanation will always fail to capture the variability that results when people are generating explanations, even if those people are generating explanations about the same system. We studied only deterministic algorithms which may be causing the models to diverge from people in how they generate hypotheses. Adding probabilistic rules may also be important for accounting for uncertainty about the parameter estimates, which in the real world are typically not given to you but must be inferred from data as well.

Note that **CET** in this case treats all explanations that sufficiently determine the observations as having equivalent rank. Because the system is deterministic, all 38 of the sufficient explanations are ranked as number 1 — or rather, because they are so numerous, number 19. This is a problem unique to **CET**, and results from its use of intervention, which ignores variables' prior distributions in determining an explanation's score.

Table 1: Rank-correlations for models and human data in Experiment 1, $P_{\text{val}} < 0.05$ in **bold**, < 0.10 in *italics*.

	Circuit		*Pearl*	
Models	ρ_{Spearman}	P_{val}	ρ_{Spearman}	P_{val}
MPE	-0.06	0.631	0.32	0.449
MRE	*0.20*	*0.074*	**0.83**	**0.017**
ET	0.08	0.460	0.17	0.700
$\mathbf{ET_{tree}}$	0.01	0.900	0.41	0.310
CET	*0.22*	*0.055*	**0.93**	**0.003**
$\mathbf{CET_{tree}}$	0.06	0.590	**0.77**	**0.032**

5 Experiment 2: Evaluation

In Experiment 1, we found evidence that at least some of the proposed models capture people's explanatory intuitions. Of course we should have expected some of the models to perform well; what is remarkable is how poorly some of the models did. In particular, we saw surprisingly poor performance from the tree-growth models as compared to their exhaustive-search evaluative counterparts.

Generating explanation is harder than only evaluating them — generation requires searching through the hypothesis set and then evaluating the generated explanations, while evaluation only requires computing a known evaluation function. The tree versions of the tree models are designed to make generation tractable. However, if complexity were the primary hurdle, in *Circuit* where the hypothesis space was much larger, we would expect tree methods to perform comparatively better than in *Pearl*. But they were relatively *worse*. This was due to the fact that the tree models were guaranteed to cut off at least 40% of participants since A and BC were the top choices, and cannot be reached in the same tree.

It is striking that methods that relied on probability (**MPE** and **ET**) performed so poorly in contrast to **MRE** and **CET**. However, these results may only apply to situations in which explanations are generated; explanations with large absolute probabilities may be difficult to access when generating explanations but could still be preferred if people only need to evaluate predefined hypotheses. There are many cases in which a hypothesis proves incredibly hard to generate, but once generated quickly becomes welcomed as the best explanation for many phenomena (e.g., Newton's and Einstein's physics). And, if conquering search problem is one of the driving factors behind the success of **MRE** and **CET**, it is possible that they could fail in the evaluation case.

In order to test these ideas, we conduct an experiment that is almost identical to Experiment 1. But, rather than asking people to generate explanations, we take that burden off of their shoulders. Instead, we ask them to evaluate a set of explanations that we generate for them.

5.1 Participants

A total of 245 participants were recruited through Amazon Mechanical Turk, with 9.8% excluded for failing to provide consent or otherwise complete the study and 25.3% excluded for failing one or more reading checks. This left 165 participants for analysis ($M(\text{age}) = 31.3, \%\text{-Female} = 34\%$): 46 in the disease version of *Circuit*, 46 in the lake version of *Circuit*, 34 in the disease version of *Pearl*, and 39 in the lake version of *Pearl*.

5.2 Stimuli

An explanation was included in the study if either criterion held:

- The explanation was generated by more than one participant in any one condition in Experiment 1.
- The explanation was in the top two explanations generated by any of the models.[5]

This yielded thirteen explanations for the *Circuit* causal structure and six for the *Pearl* causal structure.

5.3 Procedure

The materials and methods were nearly identical to those in Experiment 1, with the following important change: instead of providing an explanation, participants were asked to rate the quality of several provided explanations. Specifically, they were asked to rate each explanation "by placing the slider next to each explanation along the spectrum from Very Bad Explanation (furthest to the left) to Very Good Explanation (furthest to the right)," where intermediate ratings could fall anywhere in between.

Although the sliders were not presented with a numbering, positions implicitly corresponded to values between 0 and 100. Based on these ratings we can again create an explanation ranking for each participant, with ties being treated as in Experiment 1 as a repeated average value. By using ranks rather than continuous ratings we need only assume that participants have a monotonic relationship between bad and good, and avoid making assumptions about the particular nature of that scale for each participant.

5.4 Assessing model predictions

For each model, we calculated the scores assigned to the explanations that were provided to human participants. Because we were interested in explanation evaluation, we did not limit the ranks derived from **CET** or **ET** to those generated by the trees, but we did limit **MPE** to complete assignments, as otherwise it would be equivalent to **ET**.

To generate scores indicating the quality of each model, we created a set of intersection proportions. To illustrate, were we to consider only a single participant, this involves the following process. We take the human ranking as the veridical ranking. We then check whether the model's top rank explanation is the same as the participant's. We then check whether the model's two highest-ranked explanations are included in either of the two highest-ranked human explanations. We continue to do this for the whole explanation set, identifying the number of model explanations that were ranked at a level less than or equal to each level of human ranking. We can repeat this with every participant, to obtain the number of explanations matched at each rank for each participant. We can then take the average of these scores at each rank, giving us the intersection size for the full population.

It is important to note that the absolute intersection size is less useful than the proportion when we are comparing between causal structures. We can transform these values into intersection proportions by dividing each value by the total number of model explanations. This maps to a measure of how many of the model's top explanations are thought by the models to be at least as good as those generated by the average person up to that point.

To illustrate, suppose that we had explanation set $\mathcal{H}$: A, BC, BD, $ABCD$, and B, and we were con-

[5]Because there are many ways one can interpret what counts as one of the two "top" explanations, we allowed the top two as defined by *any* interpretation found in the literature of how to rank a model's results. For example, Yuan [2009] and Yuan et al. [2011] include only minimal explanations (i.e., explanations for which no subset has appeared prior to it in the ranking of explanations) when determining the results of **MRE**, whereas Nielsen et al. [2008] simply listed explanations based on their scores regardless of their minimal or non-minimal status.

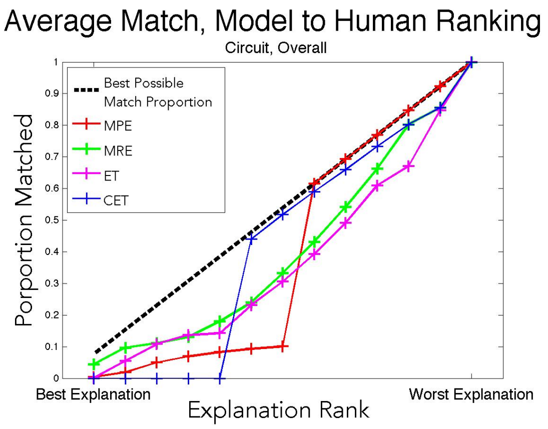


Figure 2: Results for Experiment 2: Average intersection proportions for *Circuit* conditions.

sidering a participant(P) with a ranking of $P(1) = BC$, $P(2) = A$, $P(3) = ABCD$, $P(4) = BD$, and $P(5) = B$. To compute a model's performance, we would look at the ranking that the model(M) assigned to the different explanations. If their top ranks matched, i.e., $M(1) = BC$ was the model's top choice, then the first value would be $V(M, P, 1) = \frac{1}{5} = \frac{|M(1)=\{BC\} \cap P(1)=\{BC\}|}{|\mathcal{H}|}$, and if it was not the score would be 0 since $M(1) \cap P(1) = \emptyset$. This process would be repeated for the first and second values, i.e., the next value is $V(M, P, 2) = \frac{|\{M(1)M(2)\} \cap \{P(1)P(2)\}|}{|\mathcal{H}|}$, and so on until we got to $V(M, P, 5)$ which will necessarily equal 1 since both rankings were defined relative to the same set, meaning the two sets are equivalent and are also both equivalent to $\mathcal{H}$.

Figure 2 displays the intersection proportion for the *Circuit* structure, and Figure 3 displays those for the *Pearl* structure.

Another method for capturing overall model performance is to take the sum of the average values at each point. The best one can do in the intersection proportion is to match every explanation up to that rank at each rank. A perfect summary score is, $\sum_{i=1}^{|\mathcal{H}|} i/|\mathcal{H}|$. For *Circuit* the maximum summed intersection value is $\sum_{i=1}^{13} i/13 = 7$ and for *Pearl* it is $\sum_{i=1}^{6} i/6 = 3.5$.[6] These values can be found in Table 2.

5.5 Results and discussion

As you can see in Figures 2 and 3, both **MRE** and **CET** are closer to the dotted line in general, i.e., they are better on average than either **MPE** or **ET**.

One interesting pattern to note is a trend that echoes results for **CET** in Experiment 1. **CET** stays flat

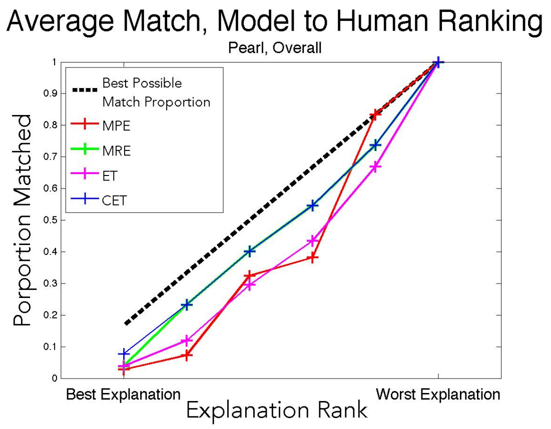


Figure 3: Results for Experiment 2: Average intersection proportions for *Pearl* conditions.

at zero for a while and then rapidly accelerates as it goes forward. This is a consequence of the interaction between **CET**'s reliance on intervention and the deterministic causal system in the *Circuit* condition. Because so many of the explanations are sufficient for bringing about the effect in question, many explanations share the role of the 'best' explanation. And because we choose an explanation's rank in the case of a tie as the average rank of all those in the tie had they not been in a tie, many of the best explanations are given a fairly high value. Thus, once we get to the sixth item, $M(1)$-$M(5)$ have had equal scores to $M(6)$, and once the values pass that threshold **CET**'s $V(M, P, \cdot)$ rapidly catches up to and passes **MRE**'s (which was otherwise in the lead). **MPE**, on the other hand, has the opposite problem: only two of its values are defined and so the other eleven explanations all receive a score of 8, resulting in perfect performance from 8 onwards (though most of its ranks are, by definition, undefined).

Table 2 shows that in both structures **CET** does the best, followed by **MRE**, then **MPE** and finally **ET**.

6 General discussion

We began this paper with the aim of systematically evaluating formal models of explanation against human intuitions as well as clarifying human explana-

Table 2: Summed intersection values for models.

Models	*Circuit* Score	*Pearl* Score
MPE	5.26	2.64
MRE	5.43	2.96
ET	4.99	2.55
CET	**5.60**	**3.00**
Max Value:	7	3.5

[6] One could think of this as an estimate of the area under the curve defined by the intersection proportions.

tion through the lens of computational models. We consider how our results address these aims.

6.1 Evaluating models of explanation

We find that **CET** and **MRE** provide reasonable but imperfect fits to human judgments in both the *Circuit* and *Pearl* structures, and for both explanation generation and evaluation. **MPE** and **ET** perform less well. This suggests that human explanation is not explained well by appealing to maximum posterior probability values. Instead, it seems that a measure of evidence (**MRE**) or causal information (**CET**) may better model human explanation.

These findings indicate that the algorithms used for generating explanations in the tree methods (**ET** and **CET**) fail to capture an important aspect of human intuitions about explanation — explanations that are radically different from one another (i.e., that cannot be reached by the same tree) may both be seen as valid explanations. In the generation task, the purely evaluative tree models outperformed their generative counterparts. The evaluation function seems to be quite important, but it has been emphasized less than the generation algorithm in previous work [Flores et al., 2005, Nielsen et al., 2008]. The evaluation function merits closer inspection.

Speaking generally, our work reveals the degree to which a model's objective alters that model's predictions. Our analyses highlight the problem with using hard intervention in deterministic cases. **CET** gave the same score to all 38 sufficient explanations that, presumably, we would want the model to distinguish. **MPE** and **ET** excel at doing what they were created to do, but we may wish to distinguish between their goals (which do not correspond closely to human explanation judgments) and the goals of models like **CET** and **MRE** (which do).

6.2 Bidirectional implications from human and formal explanation

These results indicate that formally characterizing the objective function implicit in human explanation may be a challenging but exceptionally useful task. The variability in how well these formal models performed demonstrates that despite seeming straightforward, how people choose a good explanation has many hidden subtleties and complexities. The good performance of **CET** and **MRE** relative to **MPE** and **ET** suggest that human explanation is likely more concerned with causal intervention or the relative quality of a hypothesis than it is with absolute judgments of posterior probability. But the alternative hypothesis set and the role of intervention have received relatively little attention in psychological research on explanation. On the other hand, simplicity was not explicitly represented in the formal models we explored (but, see De Campos et al. [2001]), but has been found to affect human explanatory judgments [Lombrozo, 2012]. Then, it is surprising that a large proportion of people explain using BC over A in the *Circuit* example, when BC is both less likely and more complex than A. Probability, simplicity, intervention and alternative hypotheses seem to weave a rather complex image — an image just asking to be unraveled.

All of the models we studied require knowing a priori the causal structure and parameterization, whereas people must infer these values from finite amounts of data. Though explanation has been tied to improved learning, we know much less about how the learning process and the processes for generating and evaluating explanations interact with one another. Additionally, developing extensions of these models that can learn from finite amounts of data will increase the expressivity of the models while also making them more able to deal with the problems that both humans and many real intelligent systems face.

6.3 Conclusion

Given that explanation plays an important role in human inductive judgments [Lombrozo, 2012], where humans still outperform artificial systems, we propose that models will benefit from a closer match to human judgments. And conversely, given that formal models need to make explicit the roles played by different parts of the explanatory problem and its solution, we propose that psychological accounts of explanation will benefit from models that precisely specify formal characteristics for what makes a good explanation. Both inquiries benefit from attending to the other. Our work, in simultaneously analyzing models of explanation from artificial intelligence and the psychology of human explanation, embodies this view.

Acknowledgements

This work was supported by a National Defense Science and Engineering Graduate Fellowship and a Berkeley Fellowship awarded to MP, grant number DRL-1056712 from the National Science Foundation to TL, and grant number IIS-0845410 from the National Science Foundation to TLG.

References

Carmen Lacave and Francisco J Díez. A review of explanation methods for Bayesian networks. *The Knowledge Engineering Review*, 17(2):107–127, 2002.

J. Pearl. *Probabilistic reasoning in intelligent systems.* Morgan Kaufmann, San Francisco, CA, 1988.

Changhe Yuan and Tsai-Ching Lu. Finding explanations in Bayesian networks. In *The 18th International Workshop on Principles of Diagnosis*, pages 414–419, 2007.

Tania Lombrozo. Explanation and abductive inference. *Oxford handbook of thinking and reasoning*, pages 260–276, 2012.

Ulf Nielsen, Jean-Philippe Pellet, and André Elisseeff. Explanation trees for causal Bayesian networks. *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2008.

M Flores, José Gámez, and Serafín Moral. Abductive inference in Bayesian networks: finding a partition of the explanation space. *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 470–470, 2005.

Changhe Yuan, Heejin Lim, and Tsai-Ching Lu. Most relevant explanation in Bayesian networks. *Journal of Artificial Intelligence Research*, 42(1):309–352, 2011.

Solomon E Shimony. Explanation, irrelevance and statistical independence. In *Proceedings of the ninth National conference on Artificial intelligence-Volume 1*, pages 482–487, 1991.

Luis M De Campos, Jose A Gámez, and Serafín Moral. Simplifying explanations in Bayesian belief networks. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(04):461–489, 2001.

Branden Fitelson. Likelihoodism, Bayesianism, and relational confirmation. *Synthese*, 156(3):473–489, 2007.

Joshua B Tenenbaum and Thomas L Griffiths. The rational basis of representativeness. In *Proceedings of the 23rd annual conference of the Cognitive Science Society*, pages 1036–1041, 2001.

Joshua T Abbott, Katherine A Heller, Zoubin Ghahramani, and Thomas L Griffiths. Testing a Bayesian Measure of Representativeness Using a Large Image Database. In *Proceedings of the Thirty-First Annual Conference of the Cognitive Science Society*, 2012.

Nihat Ay and Daniel Polani. Information flows in causal networks. *Advances in Complex Systems*, 11 (01):17–41, 2008.

J. Pearl. *Causality: Models, reasoning and inference.* Cambridge University Press, Cambridge, UK, 2000.

Tania Lombrozo. Simplicity and probability in causal explanation. *Cognitive Psychology*, 55(3):232–257, 2007.

Changhe Yuan. Some properties of Most Relevant Explanation. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence ExaCt Workshop*, pages 118–126, 2009.

Approximation of Lorenz-Optimal Solutions in Multiobjective Markov Decision Processes

Patrice Perny
UPMC, LIP6
patrice.perny@lip6.fr

Paul Weng
UPMC, LIP6
paul.weng@lip6.fr

Judy Goldsmith
University of Kentucky
goldsmit@cs.uky.edu

Josiah P. Hanna
University of Kentucky
jpha226@g.uky.edu

Abstract

This paper is devoted to fair optimization in Multiobjective Markov Decision Processes (MOMDPs). A MOMDP is an extension of the MDP model for planning under uncertainty while trying to optimize several reward functions simultaneously. This applies to multiagent problems when rewards define individual utility functions, or in multicriteria problems when rewards refer to different features. In this setting, we study the determination of policies leading to Lorenz-non-dominated tradeoffs. Lorenz dominance is a refinement of Pareto dominance that was introduced in Social Choice for the measurement of inequalities. In this paper, we introduce methods to efficiently approximate the sets of Lorenz-non-dominated solutions of infinite-horizon, discounted MOMDPs. The approximations are polynomial-sized subsets of those solutions.

1 INTRODUCTION

Planning under uncertainty is a central problem in developing intelligent autonomous systems. This problem is often represented by a Markov Decision Process (MDP) that provides a general formal framework for optimizing decisions in dynamic systems [2, 11]. Applications of MDPs occur in contexts such as robotics, automated control, economics, and manufacturing. The MDP model is characterized by a set of possible states, a set of possible actions enabling transitions from states to states, a reward function that gives the immediate reward generated by any admissible action a is any state s, and a transition function that gives, for any state-action pair (a, s), the resulting probability distribution over states. In such problems, the aim is to identify an optimal policy, i.e., a sequence of decision rules giving, at any stage of the process, and in any state, the action that must be selected so as to maximize the expected discounted reward over the long run.

However, there are various planning contexts in which the value of a policy must be assessed with respect to different point of views (individual utilities, criteria) and is not necessarily representable by a single reward function. This is the case in multiagent planning problems [3, 9] where every agent may have its own value system and its own reward function. This is also the case of multiobjective problems [1, 15, 4], for example path-planning problems under uncertainty when distance, travel time, and energy consumption are to be minimized simultaneously.

In such problems, resorting to n distinct reward functions is natural, so as to express utilities of actions with respect to the different objectives. Hence, MDPs are generalized into MOMDPs (Multiobjective Markov Decision Processes) by extending the reward function to map a state-action pair to a reward vector which assigns a scalar reward for each objective. The value function will also be vector-valued, and the Bellman equation will continue to define the value of a policy in all states [27]. Note that a policy that maximizes on one objective will not necessarily do the same for another. Some policies will favor one objective, some another objective, and some will be balanced towards all objectives. Even when the reward functions could be aggregated linearly, keeping them separate enables a better control of tradeoffs and better recommendation possibilities. This explains the current interest for multiobjective (multicriteria or multiagent) extensions of Markov Decision Processes in the literature [15, 4, 3, 9, 16].

When several objectives must be optimized simultaneously, most of the studies on Markov Decision Processes concentrate on the determination of the entire set of Pareto optimal feasible tradeoffs, i.e., reward vectors (corresponding to feasible policies) that cannot be improved on one objective without being downgraded on another objective. However, when randomized policies are allowed, there are infinitely many such policies. Furthermore, when only deterministic policies are allowed, there are instances of MDPs in which the size of the Pareto set grows exponentially with the number of states, thus making its exact determination intractable.

In practical cases however, there is generally no need to determine the entire set of Pareto-optimal feasible tradeoffs, but only a reduced sample of solutions representative of the diversity of feasible tradeoffs. For this reason, some authors propose to work on the determination of a polynomially sized approximation of the Pareto set covering within a given threshold all feasible tradeoffs [19]. When richer preference information is available, an alternative approach consists in optimizing a scalarizing function measuring the value of any feasible reward vector [25]. In multicriteria optimization, the scalarizing function can be any preference aggregation function monotonically increasing with Pareto dominance or any measure of the distance to a given target in the space of criteria (reference point approach, [28]). In multiple agents problems, the scalarizing function can be any Social Welfare Function aggregating individual rewards.

In this paper we propose a third approach that consists in focusing the search on Lorenz-optimal tradeoffs, i.e., Pareto-optimal tradeoffs achieving a fair sharing of rewards among objectives. Lorenz dominance (L-dominance for short) is a partial preference order refining Pareto-dominance while including an idea of fairness in preferences. It is used for the measurement of inequalities in mathematical economics [24], for example to compare income distributions over a population. In our context, it can be used to compare reward vectors by inspecting how they distribute rewards over components. L-dominance is grounded on an axiomatic principle stating that any policy modification that induces a reward transfer reducing inequalities in the satisfaction of objectives will improve the solution. Within the Pareto-set, the subset of Lorenz-optimal solutions deserve special attention because it includes all tradeoffs of interest provided a balanced reward vector is sought. Moreover, the definition of Lorenz dominance does not require any specific preference information (neither weights nor target tradeoffs), beyond the fact that there is a preference for fair solutions.

The paper is organized as follows: in Section 2 we introduce basic concepts for MOMDPs, Pareto optimality and Lorenz optimality. Section 3 presents approximate optimality concepts for multiobjective problems and establishes preliminary results concerning the construction of minimal approximation of L-optimal tradeoffs. In Section 4, we describe a general method based on linear programming to approximate the set of L-optimal solutions in the case of n objectives ($n \geq 2$) and a greedy algorithm to find approximation of minimal cardinality in the bi-objective case. Finally, in Section 5 we describe numerical tests on random instances of MDPs showing the efficiency of the proposed approaches.

2 BACKGROUND

2.1 MARKOV DECISION PROCESSES

A *Markov Decision Process (MDP)* is a tuple $\langle S, A, p, r\rangle$ where: S is a finite set of states, A is a finite set of actions, $p : S \times A \times S \to [0, 1]$ is a transition function giving, for each state and action the probability of reaching a next state, and $r : S \times A \to \mathbb{R}$ is a reward function giving the immediate reward for executing a given action in a given state [22].

Solving an MDP amounts to finding a *policy*, i.e., determining which action to choose in each state, which maximizes a performance measure. In this paper, we focus on the *expected discounted total reward* as the performance measure. A policy π is called *deterministic* if it can be defined as a function from states to actions, i.e., $\pi : S \to A$. A policy π is called *randomized* if for each state, it defines a probability distribution over actions, i.e., $\pi : S \times A \to [0, 1]$ where $\forall s, \sum_a \pi(s, a) = 1$. Note that a deterministic policy is a special case of randomized policy, i.e., $\forall s, \forall a, \pi(s, a) \in \{0, 1\}$. The expected discounted total reward for a randomized policy π in a state s can be obtained as a solution of the following equation:

$$V^\pi(s) = \sum_{a\in A} \pi(s, a)[r(s, a) + \gamma \sum_{s'} p(s, a, s')V^\pi(s')]. \tag{1}$$

Function $V^\pi : S \to \mathbb{R}$ is called the *value function* of π.

A policy whose value function is maximum in every state is an *optimal* policy. In (infinite horizon, discounted) MDPs, an optimal deterministic policy is known to exist. Such an optimal policy can be found using linear programming or dynamic programming techniques such as value iteration or policy iteration [22].

2.2 MULTIOBJECTIVE MDP

A *Multiobjective MDP* (MOMDP) is defined as an MDP with the reward function replaced by $r : S \times A \to \mathbb{R}^n$ where n is the number of criteria, $r(s, a) = (r_1(s, a), \ldots, r_n(s, a))$ and $r_i(s, a)$ is the immediate reward for objective i. Now, a policy π is valued by a value function $V^\pi : S \to \mathbb{R}^n$, which gives the expected discounted total reward vector in each state and can be computed with a vectorial version of (1) where additions and multiplications are componentwise.

Although MOMDPs can be used to solve some centralized planning problems involving multiple agents, it should not be confused with Multiagent MDPs (MMDPs) introduced in [3], which are models for coordinating agents having independent actions but a common reward function. In MOMDPs, actions are not necessarily "distributed" over agents and rewards are valued by vectors (one per agent) whereas in MMDPs, there is a single common objective and a consensus in the evaluation of states; moreover actions are distributed over agents.

To compare the value of policies in a given state s, the basic model adopted in most previous studies [8, 26, 27] is *Pareto dominance* (P-dominance for short). The *weak Pareto-dominance* is defined as follows: $\forall v, v' \in \mathbb{R}^n, v \succsim_P v' \Leftrightarrow \forall i = 1, \ldots, n, v_i \geq v'_i$ where $v = (v_1, \ldots, v_n)$ and $v' = (v'_1, \ldots, v'_n)$ and *Pareto-dominance* as: $v \succ_P v' \Leftrightarrow v \succsim_P v'$ and not$(v' \succsim_P v)$. For a set $X \subset \mathbb{R}^n$, a vector $v \in X$ is said to be *P-dominated* if there is another vector $v' \in X$ such that $v' \succ_P v$; vector v is said to be *P-optimal* is there is no vector v' such that $v' \succ_P v$. For a set $X \subset \mathbb{R}^n$, the set of *Pareto-optimal* vectors of X, called *Pareto set*, is $\mathrm{PND}(X) = \{v \in X : \forall v' \in X, \text{ not } v' \succ_P v\}$.

In MOMDPs, for a given probability distribution μ_s over initial states, a policy π is preferred to a policy π' if $\sum_s \mu_s V^\pi(s) \succ_P \sum_s \mu_s V^{\pi'}(s)$. Standard methods for MDPs can be extended to solve MOMDPs by finding Pareto-optimal policies. We recall the linear programming approach [26].

$$(\mathcal{P}_0) \quad \begin{array}{l} \max z_i = \sum_{s\in S}\sum_{a\in A} r_i(s,a)x_{sa} \quad i = 1, \ldots, n \\ \sum_{a\in A} x_{sa} - \gamma \sum_{s'\in S}\sum_{a\in A} x_{s'a}p(s',a,s) = \mu_s \ \forall s \in S \\ x_{sa} \geq 0 \quad \forall s \in S, \forall a \in A \end{array}$$

Recall that there is a one-to-one mapping between variables (x_{sa}) satisfying constraints of $\mathcal{P}_0$ and randomized policies π (i.e., $\pi(s,a) = x_{sa}/\sum_a x_{sa}$) and $\sum_{s\in S}\sum_{a\in A} R_i(s,a)x(s,a) = \sum_s \mu_s V_i^\pi(s)$ for all $i = 1, \ldots, n$. More specifically, the constraints of $\mathcal{P}_0$ define a polytope whose extreme points are deterministic policies. For a deterministic policy, in every state s, x_{sa} is non-null only for one action a. Thus, solving this multiobjective linear program amounts to optimizing the objective function $\sum_{s\in S} \mu_s V(s)$, called the *value vector* and interpreted as the expectation of a vector value function V w.r.t. probability distribution μ_s.

Following [7], one could add the following constraints to this linear program, obtaining then a mixed linear program with $0, 1$ variables, to restrict the search to deterministic policies only:

$$\begin{array}{ll} \sum_{a\in A} d_{sa} \leq 1 & \forall s \in S \\ (1-\gamma)x_{sa} \leq d_{sa} & \forall s \in S, \forall a \in A \\ d_{sa} \in \{0,1\} & \forall s \in S, \forall a \in A. \end{array} \qquad (2)$$

As Pareto dominance is a partial relation, there generally exist many Pareto-optimal policies. In fact, in the worst case, it may happen that the number of Pareto-optimal value vectors corresponding to deterministic policies is exponential in the number of states as shown in the following example, adapted from [10].

Example 1 *Let $N > 0$. Consider the following deterministic MOMDP represented in Figure 1. It has $N+1$ states. In each state, two actions (Up or Down) are possible except in the absorbing state N. The rewards are given next to the arcs representing the two actions. Here, we can take $\gamma = 1$ as state N is absorbing. In this example, there are 2^{N+1} stationary deterministic policies. Stationary deterministic policies that only differ from one another on the choice of the action in the last state N have the same value functions as the reward and the transition in those states for both actions are identical. In the initial state 0, the remaining policies induce 2^N different valuation vectors, of the form $(x, 2^N - 1 - x)$ for $x = 0, 1, \ldots, 2^N - 1$. Those different vectors are in fact all Pareto-optimal as they are on the line $x + y = 2^N - 1$.*

This example suggests that computing all Pareto-optimal solutions is not feasible in the general case. Moreover, deciding whether there exists a deterministic policy whose value vector P-dominates a given vector is known to be NP-hard [4, 23].

In this paper, we want to determine a subset of the Pareto set containing only policies that fairly distribute rewards among agents. The aim of generating well-balanced solutions has been tackled with scalararizing functions such as max-min [18, 12], augmented Tchebycheff norm [21] or WOWA of regrets [16]. However, each of these criteria focuses on a very specific idea of fairness and can only be justified when we have a very precise preferential information. A more cautious approach is to rely on Lorenz domi-

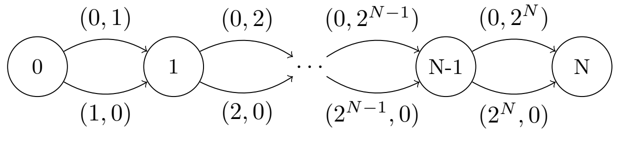


Figure 1: An instance where all deterministic policies have distinct Pareto-optimal value vectors

nance that is a partial order, leaving room for various optimally fair solutions. Let us now introduce more precisely the notions of Lorenz dominance and Lorenz optimality.

Lorenz dominance relies on a cautious idea of fairness, namely the *transfer principle*: Let $v \in \mathbb{R}^n_+$ such that $v_i > v_j$ for some i, j. Then for all $\varepsilon > 0$ such that $\varepsilon \leq v_i - v_j$, any vector of the form $v - \varepsilon e_i + \varepsilon e_j$ is preferred to v, where e_i (resp. e_j) is the vector whose i^{th} (resp. j^{th}) component equals 1, all others being 0.

This principle captures the idea of fairness as follows: If $v_i > v_j$ for some value vector $v \in \mathbb{R}^n_+$, slightly improving component v_j to the detriment of v_i while keeping the sum unchanged would produce a better distribution of rewards, and consequently a more suitable solution. Such transfers reducing inequalities are named *admissible* transfers also known as *Pigou-Dalton transfers* in Social Choice Theory. For example, value vector $(10, 10)$ should be preferred to $(14, 6)$ because there is an admissible transfer of size 4. Note that using a similar transfer of size greater than 8 would be counterproductive because it would increase inequalities in satisfaction. This explains why the transfers must have a size $\varepsilon \leq v_i - v_j$.

The transfer principle provides arguments to discriminate between vectors having the same average rewards. When combined with Pareto monotonicity (compatibility of preference with P-dominance), it becomes more powerful. For example, consider value vectors $(11, 11)$ and $(12, 9)$ respectively, we can remark that on the one hand, $(11, 11)$ is better than $(11, 10)$ due to Pareto dominance and $(11, 10)$ is better than $(12, 9)$ thanks to the Transfer Principle. Hence, we are able to conclude that $(11, 11)$ is better than $(12, 9)$ by transitivity. In order to better characterize those vectors that can be compared using improving sequences based on P-dominance and admissible transfers, we recall the definition of Lorenz vectors and Lorenz dominance (for more details see e.g. [14, 24]):

Definition 1 *For all $v \in \mathbb{R}^n_+$, the* Lorenz Vector *associated to v is the vector:*

$$L(v) = (v_{(1)}, v_{(1)} + v_{(2)}, \ldots, v_{(1)} + v_{(2)} + \ldots + v_{(n)})$$

where $v_{(1)} \leq v_{(2)} \leq \ldots \leq v_{(n)}$ represents the components of v sorted by increasing order. The k^{th} component of $L(v)$ is $L_k(v) = \sum_{i=1}^{k} v_{(i)}$.

Definition 2 *Hence, the* Lorenz dominance *relation (L-dominance for short) on $\mathbb{R}^n_+$ is defined by:*

$$\forall v, v' \in \mathbb{R}^n_+, \; v \succsim_L v' \iff L(v) \succsim_P L(v')$$

Its asymmetric part is defined by:

$$v \succ_L v' \iff L(v) \succ_P L(v').$$

Within a set X, any element v is said to be *L-dominated* when $v' \succ_L v$ for some v' in X, and *L-optimal* when there is no v' in X such that $v' \succ_L v$. The set of L-optimal elements in X, called the *Lorenz* set, is denoted $\text{LND}(X)$. In order to establish the link between Lorenz dominance and preferences satisfying the combination of P-Monotonocity and the transfer principle we recall a result of [5]:

Theorem 1 *For any pair of vectors $v, v' \in \mathbb{R}^n_+$, if $v \succ_P v'$, or if v is obtained from v' by a Pigou-Dalton transfer, then $v \succ_L v'$. Conversely, if $v \succ_L v'$, then there exists a sequence of admissible transfers and/or Pareto-improvements to transform v' into v.*

This theorem establishes Lorenz dominance as the minimal transitive relation (with respect to inclusion) satisfying compatibility with P-dominance and the transfer principle. As a consequence, the subset of L-optimal value vectors appears as a very natural solution concept in fair optimization problems. A consequence of Theorem 1 is that $v \succ_P v'$ implies $v \succ_L v'$. Hence, L-dominance is a refinement of P-dominance and the set of L-optimal vectors is included in the set of P-optimal vectors.

The number of Lorenz-optimal tradeoffs is often significantly smaller than the number of Pareto-optimal tradeoffs. For instance in Example 1, while there is an exponential number of Pareto-optimal policies, with distinct value vectors, there are only two Lorenz-optimal policies with value vectors $(\lfloor \frac{2^N-1}{2} \rfloor, \lceil \frac{2^N-1}{2} \rceil)$ and $(\lceil \frac{2^N-1}{2} \rceil, \lfloor \frac{2^N-1}{2} \rfloor)$. Unfortunately, there exist instances where the number of Lorenz-optimal value vectors corresponding to deterministic policies is exponential in the number of states, as shown in the following example adapted from [20].

Example 2 *Let $N > 0$. Consider the following deterministic MOMDP represented in Figure 2, which is an adaptation of Example 1. It has $N + 1$ states. In each state, two actions (Up or Down) are possible except in the absorbing state N. The rewards are given next to the arcs representing the two actions. Here, we can take $\gamma = 1$ as state N is absorbing.*

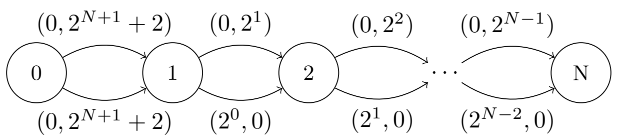


Figure 2: An instance where all deterministic policies have distinct Lorenz-optimal value vectors

From state 1, all the possible value vectors are in the set $\{(x, 2(2^{N-1} - 1 - x)) | x = 0, 1, 2, \ldots, 2^{N-1} - 1\}$. Then, from the initial state s_0, the possible value vec-

tors are in $\{(x, 3\times 2^N - 2x) | x = 0,1,2,\dots,2^{N-1}-1\}$. *The set of Lorenz vectors is then* $\{(x, 3\times 2^N - x) | x = 0,1,2,\dots,2^{N-1}-1\}$, *implying that all the Lorenz vectors are Pareto-optimal.*

This example shows that, although more discriminating than Pareto dominance, Lorenz dominance might leave many solutions incomparable. Therefore, on large instances, it may be infeasible to determine all Lorenz non-dominated solutions. Moreover, in deterministic MOMDPs, deciding whether there exists a policy whose value vector L-dominates a fixed vector is NP-hard [20].

3 APPROXIMATION OF PARETO AND LORENZ SETS

3.1 ε-COVERING OF NON-DOMINATED ELEMENTS

The examples provided at the end of Section 2 show that, even if we restrict ourselves to deterministic policies and two objectives, the set of P-optimal tradeoffs and the set of L-optimal tradeoffs may be very large. Their cardinality may grow exponentially with the number of states. Hence one cannot expect to find efficient algorithms to generate these sets exactly. This suggests that relaxing the notion of L-dominance (resp. P-dominance) to approximate the Lorenz set (resp. the Pareto set) with performance guarantees on the approximation would be a good alternative in practice. We first recall some definitions used to approximate dominance and optimality concepts in multiobjective optimization. We then investigate the construction of an approximation of the set of L-optimal tradeoffs. First, we consider the notion of ε-dominance defined as follows [19, 13]:

Definition 3 *For any* $\varepsilon > 0$, *the* ε*-dominance relation is defined on value vectors of* $\mathbb{R}^n$ *as follows:*

$$x \succsim_P^\varepsilon y \Leftrightarrow [\forall i \in N, (1+\varepsilon)x_i \geq y_i].$$

Hence we can define the notion of approximation of the Pareto set as follows:

Definition 4 *For any* $\varepsilon > 0$ *and any set* $X \subseteq \mathbb{R}^n$ *of bounded value vectors, a subset* $Y \subseteq X$ *is said to be an* ε*-covering of* $PND(X)$ *if* $\forall x \in PND(X)$, $\exists y \in Y : y \succsim_P^\varepsilon x$.

For example, on the left part of Figure 3, the five black points form an ε-covering of the Pareto set. Indeed, dotted lines define 5 cones delimiting the areas where value vectors are ε-dominated by a black point. One can see that the union of these cones covers all feasible value vectors. Of course, a given set X of feasible

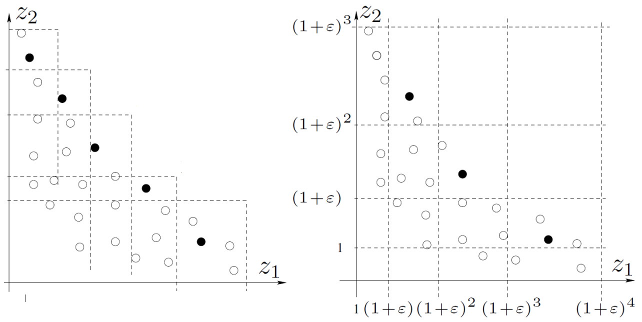


Figure 3: ε-coverings of the Pareto set

tradeoffs may include multiple ε-covering sets, set X is itself an ε-covering of X. In practice, we are interested in finding an ε-covering the size of which is polynomially bounded.

The strength of the ε-covering concept is derived from the following result of Papadimitriou and Yannakakis [19]: for any fixed number of criteria $n > 1$, for any finite $\epsilon > 0$ and any set X of bounded value vectors such that $0 < x_i \leq K$ for all $i \in N$, there exists in X an ε-covering of the Pareto set PND(X) the size of which is polynomial in $\log K$ and $1/\epsilon$. The result can be simply explained as follows: to any reward vector $x \in \mathbb{Z}^n$, we can assign vector $\varphi(x)$ the components of which are $\varphi(x_i) = \lceil \frac{\log x_i}{\log(1+\varepsilon)} \rceil$. Due to the scaling and rounding operation, the number of different possible values for φ is bounded on each axis by $\lceil \log K / \log(1+\varepsilon)\rceil$. Hence the cardinality of set $\varphi(X) = \{\varphi(x), x \in X\}$ is upper bounded by $\lceil \log K / \log(1+\varepsilon)\rceil^n$.

This can easily be illustrated using the right part of Figure 3 representing a logarithmic grid in the space of criteria. Any square of the grid represents a different class of value vectors having the same image through φ. Any vector belonging to a given square covers any other element of the square in terms of $\succsim_P^\varepsilon$. Hence, choosing one representative in each square, we cover the entire set X. If $n > 2$, squares become hypercubes. The size of the covering is bounded by the number of hypercubes in the hypergrid which is $\lceil \log K / \log(1+\varepsilon)\rceil^n$. The covering can easily be refined by keeping only the elements of PND$(\varphi(X))$ due to the following proposition:

Proposition 1 $\forall x, y \in X, \varphi(x) \succsim_P \varphi(y) \Rightarrow x \succsim_P^\varepsilon y$.

Hence, remarking that for any fixed $x_1, \dots, x_{n-1}$ there is no more than one Pareto-optimal element in vectors $\{\varphi(x_1, \dots, x_{n-1}, z), z \in \mathbb{R}\}$ the ε-covering set will include at most $\lceil \log K / \log(1+\varepsilon)\rceil^{n-1}$ elements. In Example 1 where $n = 2$, if we consider the instance with 21 states, the Pareto set contains more than one million elements (2^{20}) whereas $\lceil \log 2^{20} / \log 1.1\rceil = 146$ elements are sufficient to cover this set with a tolerance of 10% ($\varepsilon = 0.1$).

Similarly, we can define the notion of approximation of the Lorenz set as follows:

Definition 5 *For any $\varepsilon > 0$ and any set $X \subseteq \mathbb{R}^n$ of bounded value vectors, a subset $Y \subseteq X$ is said to be an ε-covering of LND(X) if $\forall x \in LND(X)\ \exists y \in Y$: $y \succsim_L^\varepsilon x$, i.e. $L(y) \succsim_P^\varepsilon L(x)$.*

In other words, Y is a ε-covering of LND(X) if $L(Y) = \{L(y), y \in Y\}$ is a ε-covering PND($L(X)$).

Hence, assuming that an algorithm A generates an ε-covering of P-optimal elements in any set X, there are two indirect ways of constructing an ε-covering of LND(X). The first way consists of computing $L(X) = \{L(x), x \in X\}$ and then calling A to determine an ε-covering of PND($L(X)$). This approach is easily implementable when the set of feasible tradeoffs X is given explicitly. Unfortunately, in the case of MOMDPs as in many other optimization problems, the feasible set X is only implicitly known. We show in Section 4 how this approach can be modified to overcome the problem in MOMDPs. A second way consists of first computing an ε-covering Y of PND(X) with A and then determine $L(Y)$ and PND($L(Y)$). This yields an ε-covering of LND(X) as shown by:

Proposition 2 *For any set X of vectors, if Y is an ε-covering of PND(X), then PND($L(Y)$) is an ε-covering of LND(X).*

Proof: For any $x \in \text{PND}(X)$, there is a $y \in Y$ such that $y \succsim_P^\varepsilon x$. Hence $(1+\varepsilon)y \succsim_P x$ and $L((1+\varepsilon)y) \succsim_P L(x)$ by Theorem 1. Since $L((1+\varepsilon)y) = (1+\varepsilon)L(y)$ we obtain $(1+\varepsilon)L(y) \succsim_P L(x)$. Also, there is $z \in \text{PND}(L(Y))$ such that $L(z) \succsim_P L(y)$ and therefore $(1+\varepsilon)L(z) \succsim_P (1+\varepsilon)L(y)$. Hence by transitivity we get $(1+\varepsilon)L(z) \succsim_P L(x)$ and therefore $L(z) \succsim_P^\varepsilon L(x)$ □

The general result of Papadimitriou and Yannakakis [19] holds for MOMDPs, as shown by Chatterjee et al. [4]. For any MOMDP $\langle S, A, p, r\rangle$ with discount factor $\gamma \in (0,1)$, for all $\varepsilon > 0$, there exists an ε-covering of Pareto-optimal tradeoffs whose size is polynomial in $|S|$ (the number of states), $|\gamma|$, $|R|$ (an upper bound on rewards), and $1/\varepsilon$, and exponential in n. Moreover, there exists an algorithm to construct an ε-covering of the Pareto set in time polynomial in $|S|, |\gamma|, |R|$, and $1/\varepsilon$ and exponential in n. This algorithm is based on a systematic inspection of the squares of the grid given in Figure 1, using linear programming techniques. Hence when the number of criteria is fixed, Proposition 2 can be used to show the existence of a fully polynomial approximation scheme (fptas) for the set of L-optimal tradeoffs:

Proposition 3 *For any fixed number of criteria $n > 1$, for any MOMDP $\langle S, A, p, r\rangle$ involving n criteria and a discount factor $\gamma \in (0,1)$, for all $\varepsilon > 0$, there exists an ε-covering of Lorenz-optimal tradeoffs whose size is polynomial in $|S|, |\gamma|, |R|$, and $1/\varepsilon$. Moreover, there exists an algorithm to construct an ε-covering of Lorenz-optimal tradeoffs in time polynomial in $|S|, |\gamma|, |R|$, and $1/\varepsilon$.*

Proof: For any fixed $n > 1$, we know that an ε-covering of the Pareto set Y of size polynomial in $|S|, |\gamma|, |R|$, and $1/\varepsilon$ can be computed in time polynomial in $|S|, |\gamma|, |R|$, and $1/\varepsilon$. Moreover, we have $\text{PND}(L(Y)) \subseteq L(Y)$ and $|L(Y)| \leq |Y|$, therefore $|\text{PND}(L(Y))|$ is polynomial in $|S|, |\gamma|, |R|$, and $1/\varepsilon$. Moreover, $L(Y)$ can be derived from Y in polynomial time and then PND($L(Y)$) is obtained from $L(Y)$ in polynomial time using pairwise comparisons. Proposition 2 concludes the proof since PND($L(Y)$) is known to form an ε-covering of L-optimal solutions. □

Proposition 2 suggests a two-phase approach: first approximate the Pareto set and then derive an approximation of the Lorenz set. In the next section we investigate more direct methods to construct an approximation of the set of Lorenz-optimal tradeoffs.

4 DIRECT CONSTRUCTIONS OF ε-COVERING OF LORENZ SET

4.1 GENERAL PROCEDURE

We now present a direct procedure for constructing an ε-covering of the Lorenz set (of bounded size), without first approximating the Pareto set. This procedure relies on the observation made in the previous section: let X be the set of feasible tradeoffs and $L(X)$ its image through the Lorenz transformation. Then PND($L(X)$), the set of P-optimal vectors in $L(X)$ is an ε-covering of the Lorenz set.

Hence to any feasible tradeoff $x \in X$, we can assign a vector $\psi(x)$ where $\psi(x)_i = \lceil \frac{\log L_i(x)}{\log(1+\varepsilon)} \rceil$. Function ψ defines a logarithmic hypergrid on $L(X)$ rather than on X. Any hypercube defined by ψ in the hypergrid represents a class of value vectors that all have the same image through ψ. Any Lorenz vector $L(x)$ belonging to a given hypercube covers any other Lorenz vector $L(y)$ of the same hypercube in terms of $\succsim_P^\varepsilon$. Hence, the original vectors x, y are such that $x \succsim_L^\varepsilon y$. Moreover the following property holds:

Proposition 4 $\forall x, y \in X, \psi(x) \succsim_P \psi(y) \Rightarrow x \succsim_L^\varepsilon y$.

Thus, we can use P-optimal ψ vectors to construct an ε-covering of the Lorenz set. Besides, due to the scaling and rounding operations, the number of different possible values for ψ is bounded on the i^{th} axis by $\lceil \log iK / \log(1+\varepsilon) \rceil$, where K is an upper bound such that $0 < x_i \leq K$. Hence the cardinality of set $\psi(X) = \{\psi(x), x \in X\}$ is upper bounded by $\Pi_{i=1}^n \lceil \log(iK)/\log(1+\varepsilon) \rceil$. Moreover, since $L_i(x) \leq L_{i+1}(x)$ we have $\psi(x)_i \leq \psi(x)_{i+1}$ for all $i = 1, \ldots, n$. Therefore, when a $\psi(x)$ does not meet this constraint the corresponding hypercube is necessarily empty and does not need to be inspected. Thus the number of hypercubes that must be inspected is at most $\Pi_{i=1}^n \lceil \log(iK)/\log(1+\varepsilon) \rceil / n! \leq \Pi_{i=1}^n \lceil i \log K / \log(1+\varepsilon) \rceil / n! \leq \Pi_{i=1}^n i \lceil \log K / \log(1+\varepsilon) \rceil / n! = \lceil \log K / \log(1+\varepsilon) \rceil^n$. Hence, by choosing one representative in each of these hypercubes, we cover the entire set $L(X)$. The size of the covering is therefore bounded by $\Pi_{i=1}^n \lceil \log(iK)/\log(1+\varepsilon) \rceil / n!$, which is smaller than $\lceil \log K / \log(1+\varepsilon) \rceil^n$. Let us consider the following example:

Example 3 *If $K = 10000$, $n = 3$ and $\varepsilon = 0.1$, the grid scanned in the Lorenz space includes $\Pi_{i=1}^n \lceil \log(iK)/\log(1+\varepsilon) \rceil / n! = 186,935$ hypercubes whereas the grid for the Pareto set includes $\lceil \log K / \log(1+\varepsilon) \rceil^n = 941,192$ hypercubes (5 times more).*

Thus, this approach is expected to be faster than the two-phase method presented in the previous section. This is confirmed by tests provided in Section 5. Moreover the resulting covering set can be reduced in polynomial time so as to keep only the elements of PND($\psi(X)$). If any hypercube can be inspected in polynomial time, this direct approach based on the grid defined in the Lorenz space provides a fptas for the set of L-optimal value vectors in MOMDPs. Let us show now how hypercubes can be inspected using linear programming.

Let z be the set of feasible value vectors $(z_1, \ldots, z_n)$ defined by $\mathcal{P}_0$ introduced in Section 2. We consider the following optimization problem designed to test whether there exists a feasible z whose Lorenz vector $L(z)$ P-dominates a given reference vector $\eta \in \mathbb{R}^n$ representing the lower corner of an hypercube in the Lorenz space.

$$(\mathcal{P}_\eta) \qquad \begin{array}{l} \max L_n(z) \\ L_k(z) \geq \eta_k, \quad k = 1, \ldots, n-1, \\ z \in Z. \end{array}$$

The objective of this optimization program is linear in variables z_i since $L_n(z) = \sum_{i=1}^n z_i$. However, none of the constraints is linear since $L_k(z)$ is the sum of the k greatest components of z which requires sorting the components for every z. Fortunately, for any fixed z, the k^{th} Lorenz component $L_k(z)$ can be defined as the solution of the following linear program [17]:

$$(\mathcal{P}_{L_k}) \qquad \begin{array}{l} \min \displaystyle\sum_{i=1}^n a_{ik} z_i \\ \begin{cases} \sum_{i=1}^n a_{ik} = k \\ a_{ik} \leq 1 \qquad i = 1 \ldots n. \end{cases} \\ a_{ik} \geq 0 \quad i = 1 \ldots n. \end{array}$$

This does not directly linearize the constraints of $\mathcal{P}_\eta$ because $\mathcal{P}_{L_k}$ is a minimization problem and consequently $\sum_{i=1}^n a_{ik} z_i \geq \eta_k$ does not imply that $L_k(z) \geq \eta_k$. Fortunately, by duality theorem, $L_k(x)$ is also the optimal value of the dual problem of $\mathcal{P}_{L_k}$:

$$(\mathcal{D}_{L_k}) \qquad \begin{array}{l} \max \quad k t_k - \displaystyle\sum_{i=1}^n b_{ik} \\ \{ \ t_k - b_{ik} \leq z_i \quad i = 1 \ldots n \\ b_{ik} \geq 0 \quad i = 1 \ldots n. \end{array}$$

Since $\mathcal{D}_{L_k}$ is a maximization problem, imposing constraint $kr_k - \sum_{i=1}^n b_{ik} \geq \eta_k$ together with the constraints of $\mathcal{D}_{L_k}$ implies that $L_k(z) \geq \eta_k$. Hence we obtain the following linear reformulation of $\mathcal{P}_\eta$:

$$(LP_\eta) \qquad \begin{array}{l} \max \displaystyle\sum_{k=1}^n z_k \\ \begin{cases} kt_k - \sum_{i=1}^n b_{ik} \geq \eta_k, \ k = 1 \ldots n-1 \\ t_k - b_{ik} \leq z_i, \quad i, k = 1 \ldots n \\ z \in Z. \end{cases} \\ b_{ik} \geq 0 \quad i, k = 1 \ldots n. \end{array}$$

Finally, if Z is the set of feasible value vectors of program $\mathcal{P}_0$ (see Section 2), then for any $p = (p_1, \ldots, p_n) \in \mathbb{N}^n$, one can test whether there exists a randomized policy, the Lorenz vector of which P-dominates vector: $\eta_\varepsilon^p = ((1+\varepsilon)^{p_1}, \ldots, (1+\varepsilon)^{p_n})$ by solving program LP'_η below with $\eta = \eta_\varepsilon^p$ and checking that the objective at optimum is greater or equal to $(1+\varepsilon)^{p_n}$.

$$(LP'_\eta) \qquad \begin{array}{l} \max \displaystyle\sum_{k=1}^n z_k \\ \begin{cases} z_k = \displaystyle\sum_{s \in S} \sum_{a \in A} r_k(s,a) x_{sa}, \ k = 1 \ldots n \\ kt_k - \sum_{i=1}^n b_{ik} \geq \eta_k, \ k = 1 \ldots n-1 \\ t_k - b_{ik} \leq z_i, \quad i, k = 1 \ldots n \end{cases} \\ b_{ik} \geq 0 \quad i, k = 1 \ldots n \\ x_{sa} \geq 0 \quad \forall s \in S, \forall a \in A \end{array}$$

Hence the whole logarithmic hypergrid in the Lorenz space can be entirely inspected using a polynomial number of calls to $LP'_{\eta_\varepsilon^p}$. One needs at most one call per integer valued vector $p \in \mathbb{N}^{n-1}$ such that

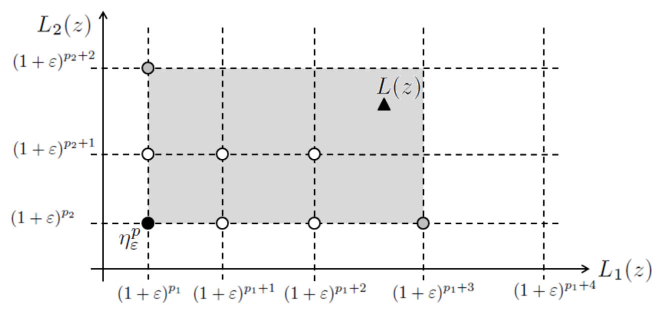

$L_2(z)$
$(1+\varepsilon)^{p_2+2}$
$(1+\varepsilon)^{p_2+1}$
$(1+\varepsilon)^{p_2}$
$L(z)$
η_ε^p
$L_1(z)$
$(1+\varepsilon)^{p_1}$
$(1+\varepsilon)^{p_1+1}$
$(1+\varepsilon)^{p_1+2}$
$(1+\varepsilon)^{p_1+3}$
$(1+\varepsilon)^{p_1+4}$

Figure 4: Grid in the Lorenz space

$p_i \leq \lceil i \log K / \log(1+\varepsilon) \rceil$ and $p_1 \leq p_2 \leq \ldots \leq p_{n-1}$. These vectors p are enumerated in lexicographic order.

This systematic inspection can be significantly sped up due to the following observation illustrated in Figure 4: let $L(z)$ be the optimal Lorenz vector obtained by solving $LP'_{\eta_\varepsilon^p}$ (represented by a triangle in Figure 4). Then none of the hypercubes corresponding to an η vector such that $L(z) \succsim_P \eta \succsim_P \eta_\varepsilon^p$ (white points in Figure 4) needs to be inspected because vectors falling in this area (colored in grey on Figure 4) are ε-dominated by $L(z)$. Hence calls to LP'_η for such η can be skipped to go directly to the next non-dominated η vectors in the grid (grey points in Figure 4).

This procedure provides an ε-covering of L-optimal randomized pure policies. Whenever we want to restrict the search to deterministic policies, a similar procedure applies, we just need to add constraints given in Equation (2) as explained in Section 2. In this case, program $LP_{\eta_\varepsilon^p}$ becomes a mix-integer linear program.

4.2 MINIMAL ε-COVERINGS: THE BIOJECTIVE CASE

The grid used to partition the entire space in the above procedure enables to avoid many unnecessary redundancies in the construction of a covering set because we keep at most one feasible policy in each hypercube. However, this procedure does not ensure that a covering of minimal cardinality will be found. In this subsection, we propose a greedy approach to generate an ε-approximation of minimal cardinality for the Lorenz set (and the Pareto set). The principle of this approach relies on a general scheme proposed in [6] for finding a minimal covering of the Pareto set in general biobjective optimization problems. Considering two objective functions z_1 and z_2, the construction consists in solving a sequence of optimization problems alternating two complementary subproblems:

Restrict-1(α_1). For any given value α_1, we want to maximize z_2 subject to the constraint $z_1 \geq \alpha_1$. The procedure returns the optimal value vector or answers *no* when no such solution exists.

Restrict-2(α_2). For any given value α_2, we want to maximize z_1 subject to the constraint $z_2 \geq \alpha_2$. The procedure returns the optimal value vector or answers *no* when no such solution exists.

Hence, the greedy construction of an ε-covering starts with the initial call v_0 =Restrict-2(0). Then we compute the following alternated sequences for $n \geq 1$:

$$\begin{aligned} u_n &= \text{Restrict-1}(v_{n-1}/(1+\varepsilon)) \\ v_{n+1} &= \text{Restrict-2}((1+\varepsilon)u_n). \end{aligned}$$

We let n increase until the feasible domain of Restrict becomes empty. Point v_0 optimizes objective 1 but does not enter into the covering set. Instead we use u_1 which is, by construction, more "central" while still covering v_0. Then, we obtain v_1 as the rightmost Pareto-optimal point on the z_1 axis that is not covered by u_1. Like v_0, v_1 does not enter into the covering. Instead we include u_2 that improves z_2 while covering v_1 and so on. The resulting set $\{u_1, \ldots, u_q\}$ provides an ε-covering set of minimal cardinality. This procedure makes only $2q$ calls to Restrict. Further details on this greedy approach and its optimality for general biobjective problems are given in [6].

The specification of procedures Restrict-1 and Restrict-2 to construct an ε-covering of the Pareto set of minimal cardinality in biobjective MDPs is straightforward from $\mathcal{P}_0$. For constructing an ε-covering of minimal cardinality for the *Lorenz set* in biobjective MDPs we solve Restrict-i(α_i) using program $LP'_i(\alpha_i)$, for i=1,2, where $LP'_i(\alpha_i)$ is a convenient adaptation of LP'_η defined as follows:

$$LP'_i(\alpha_i) \quad \begin{array}{l} \max \quad z_{3-i} \\ \left\{ \begin{array}{l} z_k = \sum_{s \in S} \sum_{a \in A} r_k(s,a) x_{sa}, \ \ k = 1,2 \\ it_i - b_{1i} - b_{2i} \geq \alpha_i, \\ t_k - b_{jk} \leq z_j, \quad j,k = 1,2 \end{array} \right. \\ b_{ik} \geq 0 \ \ i,k = 1,2 \\ x_{sa} \geq 0 \ \ \forall s \in S, \forall a \in A. \end{array}$$

Hence, Restrict-i(α_i) can be solved in polynomial time for randomized policies. Whenever we want to restrict the search to deterministic policies, we just have to add constraints given in Equation (2). In that case program LP'_i is a MIP which cannot be expected to be solved in polynomial time. It is still easily solvable by current solvers, as is shown in the next section.

5 EXPERIMENTAL RESULTS

We tested the different methods presented in this paper on random instances of MOMDPs. The rewards on each objective were randomly drawn from

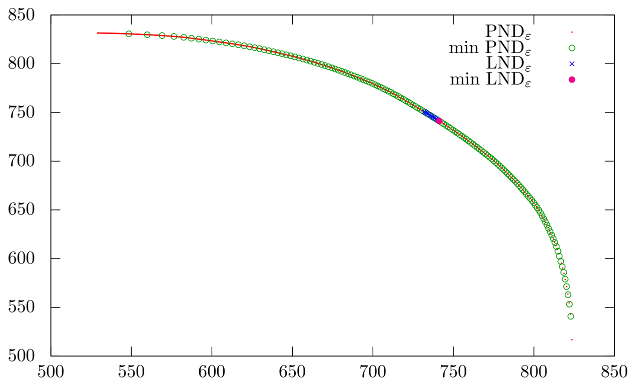


Figure 5: ε approximation of Pareto and Lorenz sets

ε	0.05	0.1	0.15	0.2
L(PND$_\varepsilon$)	265.7	169.4	126.7	101.7
LND$_\varepsilon$	5.4	4.8	4.4	4.2

Table 1: Computation times of ε-covers

$\{0, 1, \ldots, 99\}$. All the experiments were run on standard PCs with 8Gb of memory and an IntelCore 2 Duo 3.33GHz GHz processor. All LPs were solved using Gurobi 5.0. All the experimental results are averaged over 10 runs with discount factor γ set to 0.9.

First, we illustrate how the size of ε-covers can be reduced using the greedy approach both for Pareto and Lorenz. In this first series of experiments, all the instances are biojective MDPs. We set the number of states to 200 and the number of actions to 5. Parameter ε was set to 0.01. Figure 5 shows the value vectors in the objective space. PND$_\varepsilon$ (resp. LND$_\varepsilon$) is ε-cover of PND (resp. LND), min PND$_\varepsilon$ and min PND$_\varepsilon$ are the minimal ε-cover sets.

In the second series of experiments, we present the computation times (expressed in seconds) for computing the different ε-covers. In these experiments, the number of states is set to 50, the number of actions to 5 and the number of objectives to 3, and $\varepsilon \in \{0.05, 0.1, 0.15, 0.2\}$. The results are presented in Table 1 and shows that the direct approach is the most efficient. The longer computation time for the indirect method is mainly due to the determination of the approximate Pareto set.

To show the effectiveness of our approach, we computed the ε-covers for the graph presented in Example 2 with $N = 30$. The size of the ε-covers are given in Table 2.

6 CONCLUSION

We have proposed, compared and tested several efficient procedures for approximating (with performance guarantee) the set of Lorenz-optimal elements in MOMDPs. For randomized policies, the procedures presented are fully polynomial approximation schemes. Moreover for the bi-objective case, we presented a greedy approach which constructs, in polynomial time, for any given $\varepsilon > 0$, an ε-approximation of minimal cardinality of the set of Lorenz optimal tradeoffs. A similar approach works also for the Pareto set. We have shown how to modify this approach to determine covering sets using only deterministic policies. This has a computational cost since we have to solve MIPs instead of LPs. However, the numerical tests performed show that the approach remains efficient for deterministic policies on reasonably large instances. Moreover, the direct approximation of Lorenz-optimal elements enable to address larger problems than those requiring prior approximation of the Pareto set. Note that beside the `restrict` procedure, the approach is quite generic and could probably easily be adapted to other multiobjective problems.

ε	0	0.05	0.1	0.15	0.2
PND$_\varepsilon$	2^{30}	361	194	135	104
L(PND$_\varepsilon$)	2^{30}	16	8	6	4
min PND$_\varepsilon$	2^{30}	15	8	5	4
L(min PND$_\varepsilon$)	2^{30}	9	5	3	3
LND$_\varepsilon$	2^{30}	17	9	6	5
min LND$_\varepsilon$	2^{30}	4	2	2	1

Table 2: Sizes of ε-covers for the graph in Example 2

These tools provide useful information for selecting optimal actions and policies in dynamic systems. By playing with threshold ε we can increase or decrease on demand the size of our sample of solutions and provide more or less contrasted tradeoffs to cover the Pareto set or the Lorenz set. This approach could also be used to approximate f-optimal tradeoffs for any scalarizing function f monotonic with respect to Pareto or Lorenz dominance.

Beside the application to aggregation functions, another research direction would be to look for procedures to construct minimal covering sets for MOMDPs involving more than two objectives. To the best of our knowledge this remains an open problem.

Acknowledgments

Funded by the French National Research Agency under grant ANR-09-BLAN-0361; The Duncan E. Clarke Memorial Innovation Award, and NSF grants NSF EF-0850237 and CCF-1049360.

References

[1] E. Altman. *Constrained Markov Decision Processes.* CRC Press, 1999.

[2] R.E Bellman. *Dynamic programming.* Princeton university press, Princeton, 1957.

[3] C. Boutilier. Sequential optimality and coordination in multiagent systems. In *Proc. IJCAI*, 1999.

[4] K. Chatterjee, R. Majumdar, and T.A. Henzinger. Markov decision processes with multiple objectives. In *STACS*, 2006.

[5] K.M. Chong. An induction theorem for rearrangements. *Canadian Journal of Mathematics*, 28:154—160, 1976.

[6] I. Diakonikolas and M. Yannakakis. Small approximate pareto sets for biobjective shortest paths and other problems. *SIAM Journal on Computing*, 39(4):1340–1371, 2009.

[7] D. Dolgov and E. Durfee. Stationary deterministic policies for constrained MDPs with multiple rewards, costs, and discount factors. In *IJCAI*, 2005.

[8] N. Furukawa. Vector-valued markovian decision processes with countable state space. *Ann. Math. Stat.*, 36, 1965.

[9] C. Guestrin, D. Koller, and R. Parr. Multiagent planning with factored MDPs. In *NIPS*, 2001.

[10] P. Hansen. *Multiple Criteria Decision Making Theory and Application*, chapter Bicriterion Path Problems, pages 109–127. Springer, 1979.

[11] R.A. Howard. *Dynamic Programming and Markov Processes.* The M.I.T. Press, 1960.

[12] J.Y. Kwak, P. Varakantham, R. Maheswaran, M. Tambe, F. Jazizadeh, G. Kavulya, L. Klein, B. Becerik-Gerber, T. Hayes, and W. Wood. Saves: A sustainable multiagent application to conserve building energy considering occupants. In *AAMAS*, 2012.

[13] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, 10(3):263–282., 2002.

[14] A.W. Marshall and I. Olkin. *Inequalities: Theory of Majorization and its Applications.* Academic Press, 1979.

[15] A.I. Mouaddib. Multi-objective decision-theoretic path planning. *IEEE Int. Conf. Robotics and Automation*, 3:2814–2819, 2004.

[16] W. Ogryczak, P. Perny, and P. Weng. On minimizing ordered weighted regrets in multiobjective Markov decision processes. In *Int. Conf. on Algorithmic Decision Theory*, 2011.

[17] W. Ogryczak and T. Sliwinski. On solving linear programs with the ordered weighted averaging objective. *Eur. J. Operational Research*, 148:80–91, 2003.

[18] A. Osyczka. An approach to multicriterion optimization problems for engineering design. *Computer Methods in Applied Mechanics and Engineering*, 15(3):309–333, 1978.

[19] C.H. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *FOCS*, pages 86–92, 2000.

[20] P. Perny and O. Spanjaard. An axiomatic approach to robustness in search problems with multiple scenarios. In *UAI*, volume 19, pages 469–476, 2003.

[21] P. Perny and P. Weng. On finding compromise solutions in multiobjective Markov decision processes. In *European Conference on Artificial Intelligence Multidisciplinary Workshop on Advances in Preference Handling*, 2010.

[22] M.L. Puterman. *Markov decision processes: discrete stochastic dynamic programming.* Wiley, 1994.

[23] P. Serafini. Some considerations about computational complexity for multiobjective combinatorial problems. In J. Jahn and W. Krabs, editors, *Recent advances and historical development of vector optimization*, volume 294 of *Lecture Notes in Economics and Mathematical Systems*, Berlin, 1986. Springer-Verlag.

[24] A.F. Shorrocks. Ranking income distributions. *Economica*, 50:3–17, 1983.

[25] R.E. Steuer. *Multiple criteria optimization.* John Wiley, 1986.

[26] B. Viswanathan, V.V. Aggarwal, and K.P.K. Nair. Multiple criteria Markov decision processes. *TIMS Studies in the Management Sciences*, 6:263–272, 1977.

[27] D.J. White. Multi-objective infinite-horizon discounted Markov decision processes. *J. Math. Anal. Appls.*, 89:639–647, 1982.

[28] A.P. Wierzbicki. A mathematical basis for satisficing decision making. *Mathematical Mod- elling*, 3:391–405, 1982.

Solution Methods for Constrained Markov Decision Process with Continuous Probability Modulation

Janusz Marecki, Marek Petrik, Dharmashankar Subramanian
Business Analytics and Mathematical Sciences
IBM T.J. Watson Research Center
Yorktown, NY
{marecki,mpetrik,dharmash}@us.ibm.com

Abstract

We propose solution methods for previously-unsolved constrained MDPs in which actions can continuously modify the transition probabilities within some acceptable sets. While many methods have been proposed to solve regular MDPs with large state sets, there are few practical approaches for solving constrained MDPs with large action sets. In particular, we show that the continuous action sets can be replaced by their extreme points when the rewards are linear in the modulation. We also develop a tractable optimization formulation for concave reward functions and, surprisingly, also extend it to non-concave reward functions by using their concave envelopes. We evaluate the effectiveness of the approach on the problem of managing delinquencies in a portfolio of loans.

1 Introduction

This paper is motivated by the need of a loan services provider to efficiently manage a portfolio of loans in various, finite, *levels of delinquency* over a finite number of *decision periods*. In the absence of interventions, a loan is assumed to transition from one delinquency level to another across time periods according to an exogenous *base* transition probability. This transition probability can, however, be controlled by taking various intervention actions, the cost of which depends on the deviation from the base transition probability. The overall objective in managing such a portfolio of loans is to choose interventions that maximize the expected financial gain of a loan servicing operator (or equivalently to minimize its loan servicing cost), subject to some constraints on the performance of the loan portfolio in expectation. These performance constraints are motivated by both regulatory and business reasons, and are typically in terms of acceptable bounds on the *expected* percentage of loans that would result in a default (the most delinquent level) at the end of a planning horizon, or at various intermediate time periods. While we focus specifically on loans, our models and results are applicable to other domains, such as maintenance scheduling, debt collection, and marketing [1].

To determine the right sequence of such interventions, one needs to solve a stochastic dynamic decision problem. Note that it suffices to optimize the sequence of interventions independently for each loan, since all important metrics (decision-making objectives and constraints) are expressed in terms of expectations. For each decision period t we assume a finite set of states $\mathcal{S}_t$ that represent the various levels of loan delinquency for the period t. For any loan state $s_t \in \mathcal{S}_t$, let $b(s_t)$ denote the base transition probability distribution over the finite support $\mathcal{S}_{t+1}$. The decision-maker can modify $b(s_t)$ into any probability distribution $p(s_t)$ that belongs to a set $\mathcal{P}_{s_t}$ of feasible distributions. In other words, $p(s_t)$ is the modulated transition probability to other delinquency states $\mathcal{S}_{t+1}$ after an intervention corresponding to s_t. The cost of achieving this modulation is assumed to be a function of the difference, $p(s_t) - b(s_t)$.

Given the chosen interventions, let $d(s)$ represent the probability of visiting state $s \in \mathcal{S}_T$ in time period T following a sequence of $T-1$ interventions. In vector notation, we have that:

$$d = \alpha^{\mathsf{T}} P_1 \cdot P_2 \cdot \dots \cdot P_{T-1},$$

where α is an initial probability distribution over the finite set $\mathcal{S}_1$ and $P_t = [p(s_t)]_{s_t \in \mathcal{S}_t}$ is the transition probability matrix induced by the interventions. The portfolio performance constraints require that for some selected states s and values $q(s)$, $d(s) \leq q(s)$. Note that d is a complex polynomial function of the decisions p. Consequently, the total expected costs corresponding to a sequence of $T-1$ interventions and transitions, as well as the performance constraints are also non-convex polynomials of degree $|T-1|$. Because non-convex polynomial optimization problems are usually very hard to solve, this direct formulation is unlikely to lead to a tractable solution.

To derive tractable algorithms we instead cast the problem as an instance of a *constrained* Markov decision process (CMDP) [2]. The MDP states in this formulation represent the levels of a loan delinquency and the actions represent the available interventions. The performance constraints can then be conveniently represented in the CMDP framework. While CMDPs with small state and action sets can easily be formulated and solved as linear programs, the loan delinquency management problem has a continuous action set—the available interventions can continuously adjust the transition probabilities between different states. Continuous action MDPs and CMDPs have been studied extensively in terms of existence of the optimal policies, but there have been few practical computational methods proposed [2, 9]. In this paper, we propose and analyze methods for solving some specific classes of CMDPs with continuous action sets.

Continuous action spaces in the form of compact spaces $\mathcal{P}_s$ have been considered in the context of robust Markov decision problems [4, 5]. Our setting is more complex because of the constraints on state visitation probabilities and non-linear reward functions. Continuous action spaces have also been considered in the context of reinforcement learning [6, 11]. The reinforcement learning approaches, unlike the methods we propose, are only approximate and cannot easily handle state probability constraints. Finally, continuous action spaces have been also considered in recent work on Markov decision process with linear transition structure [8, 7]. However, the required linear structure is not present in the loan servicing problem. Finally, CMDPs have recently been used in optimizing the tax collection for NY state [1]. The number of actions available in the tax collection problem, however, is small and the problem can be solved using standard MDP and reinforcement learning techniques.

The remainder of the paper is organized as follows. In Section 2, we define the *finite-horizon* MDP framework with continuous action spaces and state probability constraints. In Section 3, we show that the continuous-action CMDP can be reformulated as an identical finite action CMDP under some mild assumptions. While this formulation has a finite number of actions, it may still be too large to be solved efficiently in practice. In Section 4, we show a tractable formulation of the CMDPs with *concave* reward functions as a convex mathematical optimization program. Then, Section 5 extends the convex formulation to non-concave reward functions with tractable concave envelopes. Finally, Section 6 demonstrates the efficiency of the method on a realistic loan servicing problem.

2 Framework

In this section, we first describe the basic properties of constrained Markov decision processes with continuous modulation of transition probabilities. Then, we briefly discuss a CMDP formulation of the loan management problem.

We use Δ^n to denote the probability simplex in $\mathbb{R}^n$: $\Delta^n = \{p \in \mathbb{R}^n \,:\, \mathbf{1}^\mathsf{T} p = 1\}$—this represents the set of all probability distributions over n elements. We also use $\mathbf{0}$, $\mathbf{1}$, $\mathbf{I}$ to denote a vector of all zeros, all ones, and an identity matrix respectively; their sizes are given by the context.

First, we define an abstract *finite-horizon constrained Markov decision process* (CMDP) $\mathcal{M}$ with continuously modulated transition probabilities. The finite time horizon is assumed to be: $t = 1 \ldots T$.

The finite state set at time t is denoted as $\mathcal{S}_t$ and the set of all states is $\mathcal{S} = \bigcup_{t=1\ldots T} \mathcal{S}_t$. The underlying base transitions probability from any state $s_t \in \mathcal{S}_t$ is $b(s_t) \in \Delta^{|\mathcal{S}_{t+1}|}$; that is the vector of transition probabilities from some s_t to any $s_{t+1} \in \mathcal{S}_{t+1}$, when no action is taken. The infinite continuous actions space for any $s_t \in \mathcal{S}_t$ is denoted as $\mathcal{A}(s_t)$. The set $\mathcal{A}(s_t)$ must be *compact* and satisfies $\mathcal{A}(s_t) \subseteq \Delta^{|\mathcal{S}_{t+1}|}$ and $b(s_t) \in \mathcal{A}(s_t)$. The compactness assumption ensures that all the optima are achieved. An action $a_t \in \mathcal{A}(s_t)$ for $s_t \in \mathcal{S}_t$ denotes the modulated transition probability distribution over $s_{t+1} \in \mathcal{S}_{t+1}$.

The rewards are denoted as: $r(s_t, a)$ for state s_t and action a. The initial probability distribution is: $\alpha \in \Delta^{|\mathcal{S}_1|}$. Finally, the solution must satisfy quality constraints such that the visitation probability for states in $\mathcal{Q}_i \subset \mathcal{S}$ are bounded by q_i for some indices $i \in \mathcal{I}$.

Next, we summarize the known properties of the optimal solutions of CMDPs with continuous actions. Similarly to unconstrained MDPs, there exists an optimal Markov policy π (e.g. Theorem 6.2 in [2]) under some mild assumptions, but this policy may need to be randomized. The set of randomized Markov policies $\Pi_R = \{\pi : \mathcal{S} \to \Delta^{|\mathcal{A}|}\}$. Note that the existence of an optimal policy requires that the action space is *compact*. A Markov policy is *deterministic* when the action distribution is degenerate; the set of deterministic policies is $\Pi_D = \{\pi : \mathcal{S} \to \mathcal{A}\}$.

Definition 2.1. The objective of the constrained MDP optimization is:

$$\max_{\pi \in \Pi_R} \mathbf{E}\left[\sum_{t=1}^{T-1} r(S_t, \pi(S_t))\right] \text{ s.t. } \sum_{\substack{t=1..T \\ s \in \mathcal{Q}_i}} \mathbf{P}\left[S_t = s\right] \le q_i \;,$$

for all $i \in \mathcal{I}$ where S_t are state-($\mathcal{S}_t$)-valued random variables and the constraints ensure the required solution quality.

Remark 2.2 (Uniformly optimal policies [2]). Unlike in regular MDPs, there may not be any uniformly optimal policies in a CMDP regardless of the initial state. The initial distribution is thus a key part of the CMDP definition.

In the remainder of the paper, we use sums instead of integrals to simplify the notation when using the continuous action space. Formally, one could replace all the sums by Lebesgue integrals.

For each policy $\pi \in \Pi_R$, $u_\pi(s_t, a) \in [0,1]$ denotes joint *state action visitation probability*, and $d_\pi(s_t) \in [0,1]$ denotes the *state visitation probability*. Using these terms, the return of a policy π can be written as [2]:

$$\rho(\pi) = \sum_{t=1}^{T} \sum_{\substack{s_t \in \mathcal{S}_t \\ a_t \in \mathcal{A}(s_t)}} r(s_t, a_t) \cdot u_\pi(s_t, a_t) \tag{2.1}$$

where u_π is uniquely determined by the following constraints [9]:

$$\sum_{a_t \in \mathcal{A}(s_t)} u_\pi(s_t, a_t) = d_\pi(s_t) \tag{2.2}$$

$$\sum_{s_t, a_t} u_\pi(s_t, a_t) \cdot a_t(s_{t+1}) = d_\pi(s_{t+1}) \tag{2.3}$$

$$d_\pi(s_1) = \alpha(s_1) \tag{2.4}$$

$$\frac{u_\pi(s_t, a_t)}{d_\pi(s_t)} = \pi(s_t, a_t)\ , \tag{2.5}$$

where we implicitly assume that $s_t \in \mathcal{S}_t, a_t \in \mathcal{A}(s_t), a_{t+1} \in \mathcal{A}(s_{t+1})$ in (2.3) and the constraint must hold for each t and s_{t+1}. Note that these constraint imply that $u \geq \mathbf{0}$.

The intuitive meaning of the above constraints is as follows. Constraint (2.2) requires that the state visitation probability is simply marginalized state-action visitation probability. Constraint (2.3) can be seen as a flow conservation constraint denoting that the probability of transiting to state s_{t+1} from any state s_t is equal to the probability of visiting the state. Note that a_t in (2.3) is a vector of transition probabilities. Constraint (2.4) ensures that the initial probabilities are correct and finally, Constraint (2.5) ensures that the actions are taken with the probabilities specified by the policy π.

The return in (2.1) is maximized over policies that satisfy the quality constraints of the CMDP:

$$\sum_{s \in \mathcal{Q}_i} d_\pi(s) \leq q_i$$

for all i.

In the remainder of the paper, we use $\pi(s) = a$ to denote a deterministic policy that chooses a with probability 1 and use $\pi(s, a)$ to denote a probability of taking an action a. Finally, $\pi(s)$ for a stochastic policy denotes the vector of action probabilities.

The constraints in the CMDP make it somewhat harder to solve than regular MDPs. In particular, the standard MDP solution methods, such as *value iteration* and *policy iteration* cannot be used. The main reason is that, as Remark 2.2 notes, the optimality of a policy depends on the initial distribution. Therefore, the optimal value function cannot be computed without a reference to the initial distribution. Constrained MDPs are instead solved using an extended linear program formulation of the MDP [2].

The CMDP with continuous probability modulations is even harder to solve than regular CMDPs because of the continuous action sets. In the remainder of the paper, we show how to solve the continuous-action CMDP when the reward function satisfies certain properties. In particular, if the rewards are affine the continuous-action CMDP can be reduced to one with a finite number of actions. More generally, when the rewards are concave there exists a tractable convex formulation and, surprisingly, there may exist a tractable formulation even when the rewards are non-concave.

The loan management problem can be formulated as a CMDP as follows. As mentioned above, we can formulate the evolution of each individual loan independently from other. Let the possible delinquency states be from a set $\mathcal{D}$. Assume, in addition, that the loan size is one of discrete levels from set $\mathcal{L}$; the value of loan may change as its state evolves and it is important in determining the cost of a default. The MDP states are then defined as:

$$\mathcal{S}_t = \{(t, s, l) \ : \ s \in \mathcal{D}, l \in \mathcal{L}\} \quad t = 1 \ldots T.$$

When no intervention is taken, the loan transitions between the states according to a base transition probability $b(s_t)$ for each $s_t \in \mathcal{S}_t$.

The transitions represent both the change in the delinquency state and the loan value. The interventions modify base transition probabilities to reduce the probability of the delinquency. The feasible actions we consider in our application are $\mathcal{A}(s_t) = \{p \in \Delta^{\mathcal{S}_{t+1}} \ : \ \|p - b(s_t)\|_\infty \leq \epsilon\}$—that is the difference from the base transition probability is bounded element-wise. Each intervention has a cost associated with it. The costs are convex in the scope of the transition probability modulation. In particular, we use an appropriately weighted version of $\|a - b(s_t)\|_1$ to represent the cost of action a for each state s_t. The rewards correspond to negative costs and are, therefore, concave.

3 CMDPs with Affine Rewards

In this section, we show that the continuous action sets can be replaced by finite sets when 1) the rewards are affine functions of transition probabilities, and 2) the action sets $\mathcal{A}(s)$ are polytopes for every $s \in \mathcal{S}$. In particular, we show that there exists an optimal (randomized) policy that only takes actions that correspond to the extreme points of the polytope $\mathcal{A}(s)$.

Assumption 1. The reward $r(s, a)$ is an *affine* function of $a \in \mathcal{A}(s)$ for each $s \in \mathcal{S}$:

$$r(s, a) = e_s^\mathsf{T} a + f_s\ ,$$

for some e_s and f_s.

Consider a CMDP $\mathcal{M}_1$ with continuous action sets as defined in Section 2. We can now construct a CMDP $\mathcal{M}_2$ with an identical state space to $\mathcal{M}_1$ and actions defined as:

$$\bar{\mathcal{A}}(s) = \operatorname{ext}(\mathcal{A}(s)),$$

for each $s \in \mathcal{S}$ where ext denotes the extreme points of the set. That is, the actions in $\mathcal{M}_2$ also define the actual transition probabilities as in $\mathcal{M}_1$; except the actions are restricted to the subset $\bar{A}(s)$. The reward function $\mathcal{M}_2$ is identical to the reward function in $\mathcal{M}_1$.

Theorem 3.1. *Assume that $\mathcal{A}(s)$ is a* convex polytope *and that Assumption 1 holds. Then, the optimal returns in $\mathcal{M}_1$ and $\mathcal{M}_2$ are* identical*. In addition, for any optimal policy $\pi_2^\star$ in $\mathcal{M}_2$ there exists a deterministic policy $\pi_1^\star$ in $\mathcal{M}_1$ with the same return.*

To prove Theorem 3.1 we first need to establish the existence of an optimal deterministic policy for $\mathcal{M}_1$ when the reward function is concave (or affine).

Lemma 3.2. *Assume that the function $r(s,a)$ is* concave *in a and $\mathcal{A}(s)$ is* convex *for each $s \in \mathcal{S}$. Then, there exists an optimal* deterministic *policy $\pi^\star$ in $\mathcal{M}_1$.*

Proof. Assume an optimal randomized policy $\pi_0 \in \Pi_R$; we show there exists a deterministic policy $\pi_1 \in \Pi_D$ such that $\rho(\pi_0) = \rho(\pi_1)$. The deterministic policy π_1 is constructed as:

$$\pi_1(s) = \sum_{a \in \mathcal{A}(s)} \pi_0(s,a) \cdot a,$$

for each $s \in \mathcal{S}$. Note that a is vector in this equation; that is the action π_1 is a convex combination of elements of $\mathcal{A}(s)$.

The action $\pi_1(s)$ is in $\mathcal{A}(s)$ from because this is a convex set and $\pi_1(s)$ is a convex combination of the elements of the set. Using (2.3) and (2.4) the state visitation probabilities of π_1 and π_2 are the same: $d_{\pi_0} = d_{\pi_1}$. Using this equality and the concavity of r, we have that:

$$\begin{aligned} r_{\pi_1}(s) = r(s,\pi_1(s)) &= r\Big(s, \sum_{a \in \mathcal{A}(s)} \pi_0(s,a)\cdot a\Big) \\ &\geq \sum_{a\in\mathcal{A}(s)} \pi_0(s,a) r(s,a) = r_{\pi_0}(s). \end{aligned}$$

It readily follows that the transition probabilities under π_0 and π_1 are identical and therefore $\rho(\pi_1) \geq \rho(\pi_0)$. The lemma then follows from the optimality of π_0 and from the monotonicity of the Bellman operator. The monotonicity of the Bellman operator implies that uniformly increasing the rewards also increases the return. □

Proof of Theorem 3.1. Let $\pi_i^\star$ and $\rho_i^\star$ be the optimal policy and return in $\mathcal{M}_i$ respectively. We show the equality $\rho_1^\star = \rho_2^\star$ in two steps; first, we show that $\rho_2^\star \geq \rho_1^\star$. Assume, from Lemma 3.2, that $\pi_1^\star$ is deterministic. Then, create a *randomized* policy π_2 in $\mathcal{M}_2$ such that for each $s \in \mathcal{S}$ it satisfies:

$$\sum_{\bar{a} \in \hat{\mathcal{A}}(s)} \pi_2(s,\bar{a}) \cdot \bar{a} = \pi_1^\star(s) \tag{3.1}$$

There always exists a unique π_2 that satisfies the above condition since $\hat{\mathcal{A}}(s) = \operatorname{ext}(\mathcal{A}(s))$ and $\mathcal{A}(s)$ is convex—each point in a polytope is a unique convex combination of its extreme points (e.g. Krein–Milman Theorem). The condition (3.1) guarantees that the transitions probabilities for $\pi_1^\star$ and π_2 are the same. It remains to show that the rewards for $\pi_1^\star$ and π_2 equal:

$$\begin{aligned} r_{\pi_2}(s) &= \sum_{\bar{a}\in\hat{\mathcal{A}}(s)} \pi_2(s,\bar{a}) \cdot r(s,\bar{a}) \\ &= e_s^\mathsf{T}\Big(\sum_{\bar{a}\in\hat{\mathcal{A}}(s)} \pi_2(s,\bar{a})\cdot\bar{a}\Big) + f_s = r_{\pi_1^\star}(s) \end{aligned} \tag{3.2}$$

by (3.1) of π_2 and Assumption 1. The monotonicity of the Bellman operator then implies that $\rho_2^\star \geq \rho_2 \geq \rho_1^\star$.

Next, we show that $\rho_1^\star \geq \rho_2^\star$. Let $\pi_2^\star$ be an optimal *randomized* policy in $\mathcal{M}_2$. Define a deterministic policy π_1 as follows:

$$\pi_1(s) = \sum_{\bar{a}\in\hat{\mathcal{A}}(s)} \pi_2^\star(s,\bar{a}) \cdot \bar{a}. \tag{3.3}$$

Note that (3.3) represents a convex combination of individual action vectors. It can be readily shown from (3.3) that the transition probabilities for policies $\pi_2^\star$ and π_1 are the same. Next, we show that $r_{\pi_1}(s) \geq r_{\pi_2^\star}(s)$:

$$\begin{aligned} r_{\pi_2^\star}(s) &= \sum_{\bar{a}\in\hat{\mathcal{A}}(s)} \pi_2^\star(s,\bar{a}) \cdot r(s,\bar{a}) \\ &\leq r\Big(s, \sum_{\bar{a}\in\hat{\mathcal{A}}(s)} \pi_2^\star(s,\bar{a})\cdot\bar{a}\Big) \\ &= r(a,\pi_1(s)) = r_{\pi_1}(s), \end{aligned}$$

using the concavity of the reward function. The monotonicity of the Bellman operator implies that $\rho_1^\star \geq \rho_2^\star$. This shows the required equality and the necessary policies can be constructed as defined in (3.2) and in (3.3). □

There are two main limitations of the reduction in Theorem 3.1. First, the reward function must be linear. This limitation can be easily relaxed by extending the results to rewards that are piece-wise linear and concave by considering the extreme points of the hypograph of this function. Second, even though the number of actions in this formulation is finite, it still may be very large; in the worst case, the number of the finite actions may be exponential in the number of states even when $\mathcal{A}$ are specified by a polynomial number of linear constraints. In the following sections, we resolve this limitation by directly formulating the continuous-action CMDP as a convex optimization problem.

4 CMDPs with Concave Rewards

In this section, we describe a direct formulation of the CMDP as a convex mathematical optimization problem. This formulation significantly relaxes the necessary assumptions on the MDP structure compared to Theorem 3.1 and also leads to a tractable algorithm.

We start by extending the reward function $r : \mathcal{S}_t \times \Delta^{|\mathcal{S}_{t+1}|} \to \mathbb{R}$ to $\bar{r} : \mathcal{S}_t \times \mathbb{R}_+^{|\mathcal{S}_{t+1}|} \to \mathbb{R}$ which also assigns rewards for actions that are not valid distributions. The extended function $\bar{r}(s,a)$ is defined as:

$$\bar{r}(s,a) = \mathbf{1}^\mathsf{T} a \cdot r\Big(s, \frac{a}{\mathbf{1}^\mathsf{T} a}\Big) ,$$

where $\bar{r}(s,\mathbf{0}) = 0$. Note that this function is positively homogeneous; that is $\bar{r}(s, q\cdot a) = q \cdot r(s,a)$ for $q \geq 0$. This transformation also preserves the convexity or concavity of the reward function as the following lemma states.

Lemma 4.1. *For each $s_t \in \mathcal{S}_t$, the function $\bar{f}(a) = \mathbf{1}^\mathsf{T} a \cdot f(a/\mathbf{1}^\mathsf{T} a)$ is concave (convex) on $\mathbb{R}^{|\mathcal{S}_{t+1}|}$ if and only if $f(a)$ is concave (convex) on $\Delta^{|\mathcal{S}_{t+1}|}$.*

Proof. This is a standard result which can be readily shown directly from the definition of concavity (convexity) for $q \cdot f(x/q)$ for $q \geq 0$. Assume any non-negative $\alpha + \beta = 1$, then:

$$\begin{aligned}(\alpha q_1 + \beta q_2) \cdot f\Big(\frac{\alpha x_1 + \beta x_2}{\alpha q_1 + \beta q_2}\Big) &= \\ = (\alpha q_1 + \beta q_2) \cdot f\Big(\frac{\alpha x_1 q_1}{\alpha q_1 + \beta q_2}\frac{x_1}{q_1} + \frac{\beta x_2 q_2}{\alpha q_1 + \beta q_2}\frac{x_2}{q_2}\Big) &= \\ = \alpha q_1 \cdot f\Big(\frac{x_1}{q_1}\Big) + \beta q_2 \cdot f\Big(\frac{x_2}{q_2}\Big) .\end{aligned}$$

The lemma then follows from the restriction of $q = \mathbf{1}^\mathsf{T} x$. □

Below, we show several examples of the extended function.

Example 4.2. *Assume that the reward is linear: $r(s,a) = e_s^\mathsf{T} a + f_s$. Then, the extended reward function is:*

$$\bar{r}(s,a) = e_s^\mathsf{T} a + \mathbf{1}^\mathsf{T} a \cdot f_s .$$

Example 4.3. *Assume that the reward is defined by a norm: $r(s,a) = -\|a - \bar{a}_s\|$. Then, the extended reward function is:*

$$\bar{r}(s,a) = -\|a - \mathbf{1}^\mathsf{T} a \cdot \bar{a}_s\| .$$

Example 4.4. *Assume that the reward is defined by a squared L_2 norm: $r(s,a) = -\|a - \bar{a}_s\|_2^2$. Then, the extended reward function is:*

$$\bar{r}(s,a) = -\frac{1}{\mathbf{1}^\mathsf{T} a} \cdot \|a - \mathbf{1}^\mathsf{T} a \cdot \bar{a}_s\|_2^2 .$$

We are now ready to formulate the convex optimization problem. Constrained MDPs are typically solved using a linear program formulation based on the state-action visitation probabilities u as the optimization variables [2]. Such formulation would clearly lead to a semi-infinite optimization problem because of the continuous action space and the need to have a decision variable for each state and action pair. To get a tractable formulation, we instead use decision variables $u(s_t, s_{t+1})$, which represent the *joint* probability of visiting s_t *and* transiting to s_{t+1}. State visitation probabilities $d(s_t)$ can be derived from these variables by marginalizing over s_{t+1} similar to (2.2).

The main challenge with the formulation based on the decision variables $u(s_t, s_{t+1})$ is to ensure that the corresponding transition probabilities represent feasible actions in $\mathcal{A}(s)$. We use the notation $u(s_t,\cdot)$ represents the vector of values indexed by the second argument. Then, the vector of transition probabilities from state s_t is $u(s_t,\cdot)/d(s_t)$ which must be feasible in $\mathcal{A}(s_t)$. The constraints $u(s_t,\cdot)/d(s_t) \in \mathcal{A}(s_t)$ are non-linear and non-convex in the state visitation probabilities $d(s_t)$. Therefore, a direct formulation would be non-convex and difficult to solve.

To derive a convex formulation, let $\mathcal{A}(s_t)$ be a convex set defined by *convex* constraints for $s_t \in \mathcal{S}_t$:

$$\mathcal{A}(s_t) = \{a \in \Delta^{|\mathcal{S}_{t+1}|} \;:\; f_{s_t}^j(a) \leq 0,\, j \in \mathcal{J}\} ,$$

for some $f_{s_t}^j$. The feasibility constraints on the transition probabilities that have to be satisfied by the solution u then become:

$$f_s^j\Big(\frac{u(s,\cdot)}{d(s)}\Big) \leq 0 . \tag{4.1}$$

This function is non-convex in $d(s)$ and, therefore, cannot be used to formulate a convex optimization problem. To get an identical but convex constraint, first define an extended constraint function:

$$\bar{f}_s^j(a) = \mathbf{1}^\mathsf{T} a \cdot f_s^j\Big(\frac{a}{\mathbf{1}^\mathsf{T} a}\Big) ,$$

where by definition $\bar{f}_s^j(\mathbf{0}) = 0$. Note that $d(s) = \mathbf{1}^\mathsf{T} u(s,\cdot)$. The constraint (4.1) can be multiplied by $d(s)$ to get the constraint:

$$d(s) \cdot f_s^j\Big(\frac{u(s,\cdot)}{d(s)}\Big) = \bar{f}_s^j(u(s,\cdot)) \leq 0 . \tag{4.2}$$

The function $\bar{f}_s^j$ is convex from Lemma 4.1 and the constraint (4.2) is equivalent to (4.1) since $d(s) \geq 0$ and $u(s,\cdot) = \mathbf{0}$ whenever $d(s) = 0$.

We are now ready to formulate the optimization problem that can be used to compute the optimal policy in CMDPs:

$$
\begin{aligned}
\max_{u\geq \mathbf{0}, d\geq \mathbf{0}} \quad & \sum_{s\in\mathcal{S}} \bar{r}(s,u(s,\cdot)) \\
\text{s.t.} \quad & d(s_1) = \alpha(s_1) \quad \forall s_1 \in \mathcal{S}_1 \\
& d(s_t) = \sum_{s_{t+1}} u(s_t,s_{t+1}) \\
& d(s_t) = \sum_{s_{t-1}} u(s_{t-1},s_t) \\
& \sum_{s\in\mathcal{Q}_i} d(s) \leq q_i \qquad i\in\mathcal{I} \\
& \bar{f}^j_s(u(s,\cdot)) \leq 0 \quad j\in\mathcal{J}
\end{aligned}
\tag{4.3}
$$

Each s_t is implicitly considered to be in $\mathcal{S}_t$ and each s is implicitly considered to be in $\mathcal{S}$. Note that:

$$\sum_{s\in\mathcal{S}} \bar{r}(s,u(s,\cdot)) = \sum_{s\in\mathcal{S}} d(s)\cdot r\Big(s, \frac{u(s,\cdot)}{d(s)}\Big).$$

The formulation in (4.3) reduces to a linear program when the sets of feasible actions are polytopes as the following example shows.

The intuitive meaning of the constraints in (4.3) the same as in Eqs. (2.2) to (2.5). The main difference from the standard LP formulation is the objective function, which is expressed in terms of the extended reward function, and the last constraint, which is expressed in terms of the extended action constraint functions. The optimal policy $\pi^\star$ can be extracted from the optimal solution $u^\star, d^\star$ as according to Theorem 4.6.

Example 4.5. *Assume that the set of feasible actions is a polytope for each $s_t \in \mathcal{S}_t$:*

$$\mathcal{A}(s_t) = \{a \in \Delta^{|\mathcal{S}_{t+1}|} \;:\; H_s a \leq h_s\}.$$

Then, the constraints $\bar{f}^j_s(a) \leq 0$ for all $j\in\mathcal{J}$ become:

$$
\begin{aligned}
\mathbf{1}^\mathsf{T} a \cdot H_s \frac{a}{\mathbf{1}^\mathsf{T} a} &\leq \mathbf{1}^\mathsf{T} a\cdot h_s \\
H_s a &\leq \mathbf{1}^\mathsf{T} a \cdot h_s \;,
\end{aligned}
$$

which is a set of linear constraints.

The following theorem states the correctness of the formulation (4.3).

Theorem 4.6. *Assume that, for each $s\in\mathcal{S}$, $r(s,a)$ is concave in a and the set $\mathcal{A}(s)$ is convex. Let $u^\star, d^\star$ be the optimal solution of* (4.3) *and define a* deterministic *policy π:*

$$\pi(s) = u^\star(s,\cdot)/d^\star(s).$$

That is, $\pi(s)$ maps a state to a vector of state transition probabilities. Then, π is an optimal policy and the objective value of (4.3) *equals to $\rho(\pi)$. In addition,* (4.3) *is a convex optimization problem.*

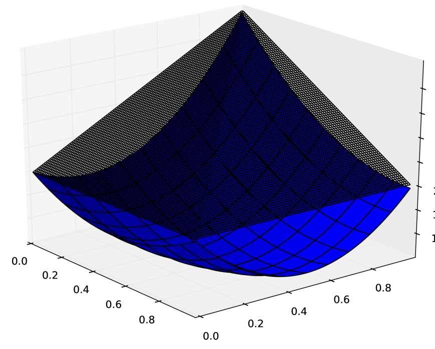


Figure 1: A convex function and its concave envelope over a unit square.

Proof. We first show that the optimal policy $\pi^\star$ is feasible in (4.3) and the corresponding objective value equals the return of the optimal policy $\pi^\star$. Given an optimal *deterministic* policy $\pi^\star$ (from Lemma 3.2), construct the solution u, d in (4.3) as $u(s_t,\cdot) = d(s_t)\cdot\pi^\star(s_t,\cdot)$. It is well known (e.g. [9]) that there is a unique such solution to all constraints without $\bar{f}^j_s(u(s,\cdot)) \leq 0$. As described above, this constraint is valid from (4.1) and (4.2) because $d \geq \mathbf{0}$. Therefore, $\sum_{s\in\mathcal{S}} \bar{r}(s,u^\star(s,\cdot)) \geq \rho(\pi^\star)$. The reverse inequality $\sum_{s\in\mathcal{S}} \bar{r}(s,u^\star(s,\cdot)) \leq \rho(\pi^\star)$ can be shown similarly by constructing a feasible policy from any solution u, d using the construction from the statement of the theorem. The convexity of the optimization problem follows readily from Lemma 4.1. $\square$

The computational complexity of solving (4.3) depends on the form of $\bar{r}$; the problem is tractable for most common concave functions. In particular, (4.3) is tractable for concave piecewise linear functions and concave quadratic functions. Note that this formulation generalizes the setting in Section 3 and has a smaller computational complexity.

5 CMDPs with Non-concave Rewards

In this section, we describe how to tractably solve CMDPs with non-concave reward functions. The approach relies on the fact that the optimal return of any constrained MDP is unaffected if the rewards are replaced by their *concave envelope* thereby obtaining a concave maximization problem.

The *concave envelope* $g(x)$ of a function $f(x)$ is defined as [3]:

$$g(x) = \sup\{t \;:\; (x,t) \in \operatorname{conv}\operatorname{hypo} f\},$$

where conv is the convex hull and hypo is the hypograph of f. A hypograph of f is defined as: $\operatorname{hypo} f = \{(x,t) \;:\; t \leq f(x)\}$. The supremum above is achieved whenever f is bounded and $\mathcal{A}(s)$ are compact, which are the assumptions

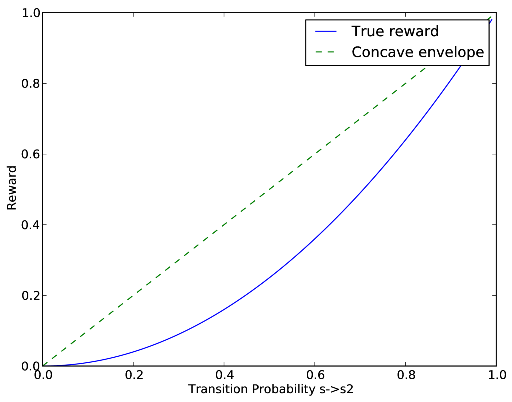


Figure 2: Example of concave envelope of the reward achievable by a randomized policy.

that we make. A concave envelope is important because it is the smallest concave function that is greater than f.

Example 5.1. *Consider a function* $f(x,y) = x^2 + 2 \cdot y^2 - x \cdot y + 2 - x - y$ *defined on the interval* $[0,1] \times [0,1]$. *The concave envelope of this convex function is the piecewise linear concave function* $g(x,y) = \min\{y+2, -x+3\}$. *Fig. 1 shows the convex function* f *and its concave envelope* g.

Assume a CMDP $\mathcal{M}$ with a reward function r and construct a CMDP $\mathcal{M}_e$ with a reward r^e that is the *concave envelope* of r for each $s \in \mathcal{S}$:

$$r^e(s,a) = \sup\{t \ : \ (x,t) \in \operatorname{conv} \operatorname{hypo} r(s,x)\},$$

where hypo is over $\mathcal{A}(s)$. Let $\rho(\pi)$ and $\rho_e(\pi)$ be the returns of π in $\mathcal{M}$ and $\mathcal{M}_e$ respectively.

The motivation for considering the concave envelope of the rewards is that the transition probabilities with this reward can be actually achieved by appropriately randomizing the policy. The following example shows this property.

Example 5.2. *Consider a state* s *with transitions to two other states* s_1 *and* s_2 *with continuous modulation of probabilities in the set* $\mathcal{A}(s) = \Delta^2$. *For any action* a, *let* a_1 *and* a_2 *represent the transition probabilities to states* s_1 *and* s_2 *respectively. Consider a* convex *reward function* $r(s,a) = a_2^2$ *and its concave envelope* $r_e(s,a) = a_2$ *depicted in Fig. 2. To show that the optimal policy will be always randomized between the extreme points, assume for example that the optimal policy is to take the transition probability* $(0.6, 0.4)$. *Directly taking an action* $(0.6, 0.4)$ *accrues a reward* $0.4^2 = 0.16$. *However, taking action* $(0,1)$ *with probability* 0.4 *and action* $(1,0)$ *with probability* 0.6 *accrues a higher reward of* 0.4. *In general, the maximal reward for each transition probability can be achieved by the maximal convex combination of other feasible actions which exactly yields the concave envelope.*

The CMDP $\mathcal{M}$ cannot be solved using (4.3) because of the non-concave rewards. On the other hand, because the rewards in $\mathcal{M}_e$ are concave, it can be easily formulated as (4.3). Note, however, that the optimal solution of $\mathcal{M}_e$ is not necessarily optimal in $\mathcal{M}$. The following theorem states that the optimal solution for $\mathcal{M}$ can be easily constructed from the optimal solution to $\mathcal{M}_e$ by appropriately randomizing between the extreme points of the concave envelope.

Theorem 5.3. *Let* $\pi_e^\star$ *be an optimal policy in CMDP* $\mathcal{M}_e$. *Then, one can construct an optimal policy* $\pi^\star$ *in* $\mathcal{M}$ *such that 1)* $\rho_e(\pi_e^\star) = \rho(\pi^\star)$ *and 2) the transition probabilities* $\pi^\star$ *and* $\pi_e^\star$ *are identical.*

Proof. First, we can assume $\pi_e^\star$ to be deterministic without loss of generality from Lemma 3.2. Clearly, we have from the optimality of $\pi_e^\star$ and from $r_e(s,a) \geq r(s,a)$ that:

$$\rho_e(\pi_e^\star) \geq \rho_e(\pi^\star) \geq \rho(\pi^\star) \ .$$

To show the equality, it only remains to show that $\rho_e(\pi_e^\star) \leq \rho(\pi^\star)$. For any $s \in \mathcal{S}$, because the value $r^e(s,\cdot)$ is a maximum in a *closed* convex hull, it is on its boundary. Therefore, for any a there exist $a_i \in \mathcal{A}(s)$ such that $r^e(s,a_i) = r(s,a_i)$ (i.e. the extreme points of the hypograph) and $\lambda_i \in [0,1]$ such that:

$$r^e(s,a) = \sum_{i=1}^{m} \lambda_i \cdot r(s,a_i),$$

such that $\lambda \geq \mathbf{0}$, $\sum_i \lambda_i = 1$, and $a = \sum_i \lambda_i \cdot a_i$. Then, construct a policy π as follows:

$$\pi(s,a_i) = \lambda_i.$$

It can be shown readily that the transition probabilities of π and $\pi_e^\star$ are the same, since $a = \sum_i \lambda_i \cdot a_i$ when assuming $a = \pi_e^\star(s)$. Then:

$$r_\pi(s) = \sum_i \lambda_i \cdot r(s,a_i) = r^e(s,a) = r^e_{\pi_e^\star}(s).$$

Therefore, the rewards and transitions of π and $\pi^\star$ are the same, which also implies $\rho_e(\pi_e^\star) \leq \rho(\pi^\star)$. □

A CMDP with non-concave rewards, therefore, can be solved as follows. First, construct a concave envelope of the rewards. Then, use (4.3) to solve the new CMDP and get a policy $\pi_e^\star$. Finally, construct the optimal $\pi^\star$ according to the construction in the proof of Theorem 5.3. That is, any action a is replaced by randomizing among actions a_i by probabilities λ_i. The points a_i depend on the construction of the concave envelope. The values λ_i can be readily computed by linear programming in general settings.

The tractability of the concave envelope approach depends on several factors. First, constructing a concave envelope is difficult in general. Second, the computed concave envelope may not have a formulation that is easily optimized. A particular case of interest is when the rewards are *convex*. Then, the concave envelope is piecewise linear and can be expressed in terms of the extreme points of $\mathcal{A}(s)$ as a linear program—it is a maximization over the convex combination of the extreme points. When the reward function is submodular on the lattice of extreme points, the envelope can be further simplified [10].

6 Application to Loan Delinquency Management

In this section, we describe the empirical results from an application of the new CMDP solution methods for both a real and a synthetic loan delinquency management problem.

We applied the proposed methods to managing the delinquencies of a loan portfolio of an actual service provider. While we are not authorized to disclose detailed results of this application, we can report the impact of our solution method. There are 8 possible states of loan delinquency; the transition probabilities can be modulated in 4 of them. The probabilities are influenced by investing resources, such as principal reduction, in the appropriate loans. The portfolio performance targets need to be achieved within a horizon of 6 months. The ranges of possible modulations and their costs were derived from corresponding transition probabilities in prior months.

The real-world empirical study was conducted to establish the necessity of a global optimization method for solving this problem. We initially evaluated a simple greedy algorithm which iteratively finds an optimal modulation of probabilities in a month t assuming that the base transitions in future months will not be modified. This greedy method returned solutions characterized by high fluctuations in monthly investments in loan servicing operations. Because the method assumes no modulations after the month t, the modulations in month t had to be overly aggressive. In the next month $t + 1$, the portfolio would be in a sufficiently good state to merit no further modulations. These month-to-month fluctuation are resource-intensive and undesirable. The optimal method proposed here smoothens out these fluctuations and can result in a significant overall reduction of resources needed to meet portfolio targets over the whole planning horizon. Experiments on six actual loan portfolios for a time horizon of six months have revealed that using the optimal method proposed in this paper has allowed for an average 13.97% reduction in the expected costs of portfolio servicing operations in comparison with the benchmark strategies used by loan managers.

Next, we proceed with an evaluation of the solution quality and scalability of the proposed algorithms on a set of synthetic loan delinquency management problems. We consider a variable number of loan delinquency states and a

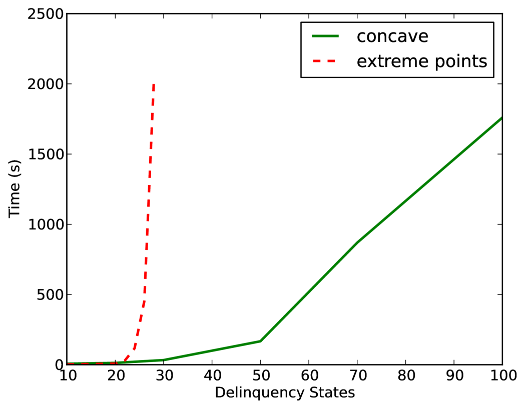


Figure 3: Time to solve a CMDP for as a function of number of states. The method "extreme points" is described in Section 3 and "concave" is described in Section 4.

fixed horizon of 6 periods. The states are ordered; the increasing order represent the increasing delinquency state of a loan, such as the number of weeks behind payments. The first state represents the loan to be current and the last state represents the default. The probability of increasing delinquency in the given period increases logarithmically with the current state of delinquency. In other words, accounts that are delinquent now are more likely to become even more delinquent in the future. The probability of the delinquency decreasing to any less delinquent state is uniform. The feasible actions are allowed to modulate any single transition probability by at most $\epsilon = 0.4$. The rewards are linear in the deviation from the base probability in each element: $-\|a - b\|_1$. The quality constraints on the probability of the default (last state) is $q = 0.04$.

Fig. 3 compares the time to solve the CMDP using the extreme points formulation described in Section 3 versus the tractable concave method described in Section 4. The timings were obtained using CPLEX 12.5 running on an Intel Core i5 1.5 GHz processor. As expected, the tractable method scales much better with the number of states. While the concave method can easily solve problems with 100s of states, the extreme-point method becomes intractable with more than 30 states. In our benchmark problem, the number of extreme points grows exponentially with the number of states. The solution quality with the two methods is identical since they are both optimal. Fig. 4 shows the sensitivity of the return to the quality constraint—the limit on the probability of a loan to end in default.

Because the rewards may often be non-concave, we also evaluate the approach assuming convex quadratic rewards $\|a - b\|_2^2$; this is relevant in particular when the economies of scale become important. We compare the algorithm from Section 5 with a simple naive approach which uses

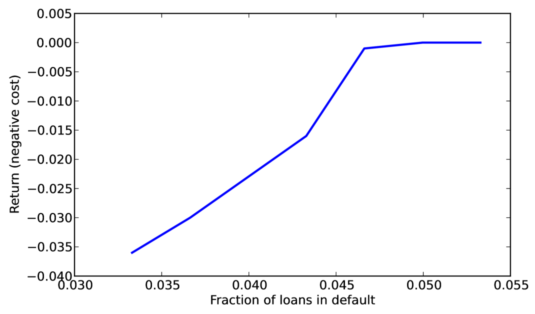


Figure 4: Return of the optimal solution as a function of the limit on the fraction of loans in default.

a linear approximation of the reward function. For the quadratic function and 30 delinquency states, the optimal method based on the concave envelope achieves a return of 0.14, while the approximation achieves return of 0. The difference in the return between these two methods can be arbitrarily large depending on the problem formulation.

7 Conclusion

We proposed three solution methods for solving constrained MDPs with continuous modulation of the probabilities. The MDP formulation was motivated by a practical need to optimally manage the delinquencies of a loan portfolio. We are not aware of any previous methods in the literature that can be used to solve this class of problems. The first method reduces the continuous action sets to finite when the rewards are affine and feasible sets polyhedral. The second formulation is a tractable optimization problem which applies to arbitrary concave reward functions. Finally, the third formulation extends the second one to non-concave rewards.

Our experimental results show that the method based on the convex optimization problem scales well and can solve problems with a large number of states in a few seconds. The method based on extreme point enumeration does not scale well, but performs better for very small problems and can be used in theoretical analysis of the result. Finally, when using the concave envelope of the rewards can significantly improve the solution quality when compared to naive approaches. While this method has not been deployed yet, the initial test results indicate that it can lead to significant improvements compared with the current greedy approach.

There are several important way in which our results can be extended. First, we considered a risk-neutral loan service provider whose utility can be expressed in terms of expectations. However, it may be desirable to extend the approach to risk-averse setting in which the service provider would be willing to trade off a higher servicing cost for a lower probability of violating the quality constraints. Other extensions involve improving the scalability of the concave envelopes for various classes of convex functions and extensions to problems with many states.

References

[1] Naoki Abe, Prem Melville, Cezar Pendus, Vince P. Thomas, James J. Bennett, Gary F. Anderson, Brent R. Cooley, Melissa Kowalczyk, Mark Domick, and Timothy Gardinier. Optimizing debt collections using constrained reinforcement learning. In *KDD*, 2010.

[2] Eitan Altman. *Constrained Markov Decision Processes*. Chapman and Hall/CRC, 1999.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] Garud N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30:257–280, 2005.

[5] Arnab Nilim and Laurent El Ghaoui. Robust Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

[6] J. Pazis and M. Lagoudakis. Binary action search for learning continuous action constrol policies. In *International Conference on Machine Learning*, pages 793–800, 2009.

[7] Marek Petrik and Dharmashankar Subramanian. An approximate solution method for large risk-averse markov decision processes. In *Proceedings of Conference on Uncertainty in Artificial Intelligence*, 2012.

[8] Marek Petrik and Shlomo Zilberstein. Linear dynamic programs for resource management. In *Conference on Artificial Intelligence*, 2011.

[9] Martin L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc., 2005.

[10] Mohit Tawarmalani, Jean-Philippe P. Richard, and Chuanhui Xiong. Explicit convex and concave envelopes through polyhedral subdivisions. *Mathematical Programming*, 2012.

[11] Ari Weinstein and Michael L. Littman. Bandit-based planning and learning in continuous action markov decision processes. In *International Conference on Automated Planning and Scheduling*, 2012.

The Supervised IBP: Neighbourhood Preserving Infinite Latent Feature Models

Novi Quadrianto
University of Cambridge
Cambridge, UK
novi.quadrianto@eng.cam.ac.uk

Viktoriia Sharmanska
IST Austria
Klosterneuburg, Austria
viktoriia.sharmanska@ist.ac.at

David A. Knowles
Stanford University
Stanford, CA, US
knowles84@gmail.com

Zoubin Ghahramani
University of Cambridge
Cambridge, UK
zoubin@eng.cam.ac.uk

Abstract

We propose a probabilistic model to infer supervised latent variables in the Hamming space from observed data. Our model allows simultaneous inference of the number of binary latent variables, and their values. The latent variables preserve neighbourhood structure of the data in a sense that objects in the same semantic concept have similar latent values, and objects in different concepts have dissimilar latent values. We formulate the supervised infinite latent variable problem based on an intuitive principle of pulling objects together if they are of the same type, and pushing them apart if they are not. We then combine this principle with a flexible Indian Buffet Process prior on the latent variables. We show that the inferred supervised latent variables can be directly used to perform a nearest neighbour search for the purpose of retrieval. We introduce a new application of dynamically extending hash codes, and show how to effectively couple the structure of the hash codes with continuously growing structure of the neighbourhood preserving infinite latent feature space.

1 Introduction

In statistical data analysis, latent variable models are used to represent components or properties of data that have not been directly observed, or to represent hidden causes that explain the observed data. In many cases, a natural representation of an object would allow each object to admit multiple latent features. Classical statistical techniques require the number of latent features to be fixed a priori. Recently, nonparametric Bayesian models have emerged as an elegant approach to deal with this issue by allowing the number of features to be inferred from data. One class of these models utilise the Indian Buffet Process (IBP) prior (Griffiths & Ghahramani, 2005) to allow an unbounded number of features. Almost all IBP-based statistical models are geared towards *unsupervised* latent feature learning. While unsupervised latent feature models are promising, for example, as an exploratory tool for discovering compact hidden structures in observed data, in many practical settings we seek *supervised* latent variables, that are semantically meaningful and encode supervised side information. Such supervised side information can be expressed as neighbours (similar) and non-neighbours (dissimilar) data pairs, as in (Schultz & Joachims, 2003) for example, and can be used for retrieval of semantically similar neighbours (Weiss et al., 2009).

This paper presents a method to simultaneously infer the dimension of the binary latent representations, and their values so as to encode supervised side information. Binary representations are very attractive for reducing storage requirements and accelerating search and retrieval in large collections of high dimensional data. In recent years, there has been a lot of interest in designing compact binary hash codes such that vectors that are similar in the original data space are mapped to similar binary strings as measured by Hamming distance (Salakhutdinov & Hinton, 2007). However, existing hashing work is typically performed in a static way, that is, a fixed number of bits has to be discovered to model data. We aim to have an approach that is flexible to add bits automatically in order to model new unseen data. This is useful for adaption of hash code to the dynamic and streaming nature of the Internet data, for example.

We present an application of our method to dynamic hash code extension in image retrieval. This application combines the merits of *non-Bayesian* methods that can handle large data (such as Weiss et al. (2009)) and *non-parametric Bayes* that allows extension of codes as required. Our goal is to utilise existing binary hash codes, and learn how to *extend* the codes *pro re nata* in a supervised way when more data become available. For a model that attempts to identify hash codes of dynamically changing data, we argue that the assumption on the number of extended bits to be fixed beforehand is unrealistic.

Our supervised latent variable model enforces latent variables associated with objects of the same semantic concept to have similar values, and latent variables associated with objects of different concepts to have dissimilar values. To achieve this, we define a likelihood function in Section 2 that views this criterion as *preference* relation. When coupled with a flexible prior on infinite sparse binary matrices and a data likelihood, we are able to characterise a probabilistic model for supervised infinite latent variables problems. For the data likelihood, we explore two directions: a standard linear Gaussian model, and our proposed linear probit dependent model, detailed in Section 3. We discuss inference in Section 4 and predictive distribution of our model in Section 5. We present experimental results, including an application in extending hash codes, in Section 6, and draw conclusions in Section 7. First, however, we give a short overview of related work to provide some context for our contributions.

1.1 Nearest Neighbour Retrieval

The majority of retrieval techniques today rely on some form of nearest neighbour search. Supervision is an integral component to improve the quality of retrieved results. This is achieved, for instance, in a query-dependent manner by analysing pairs of documents that are returned in response to the text query (Schultz & Joachims, 2003). The supervised information is used to perform metric learning, which maps the original representation of the data samples to a new, preferably low-dimensional, representation where similar samples have small Euclidean distance, and dissimilar samples are separated by a large distance.

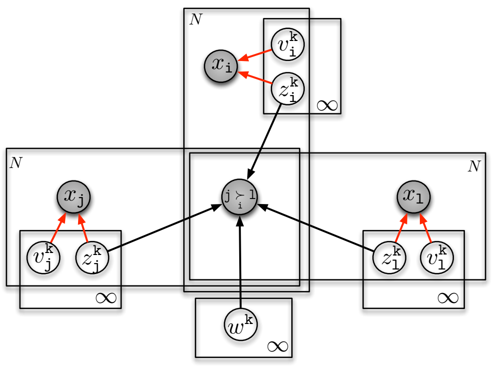
For datasets with millions or even billions of entries, even approximate nearest neighbours search techniques such as randomised neighbourhood graphs (Arya et al., 1998) and cover trees (Beygelzimer et al., 2006) are typically infeasible, and one has to resort to hashing approaches. Hash code is a short binary string that can act as an index to directly access elements in a database. Indyk & Motwani (1998) introduce locality sensitive hashing, which purely relies on randomisation techniques yet provides guarantees of preserving metric similarity for sufficiently *long* codes. Next, several machine learning methods have been developed to learn a *compact* hash code Salakhutdinov & Hinton (2007); Torralba et al. (2008); Weiss et al. (2009); Norouzi et al. (2012) among others, and to learn hash codes with better discrimination power, Mu et al. (2010); Wang et al. (2012) for example.

1.2 Distance Metric Learning

Approximate nearest neighbour search in general, and hashing-based approaches in particular, provide a powerful and well developed tool for efficient nearest neighbour retrieval from large databases. However, they typically rely on the availability of a meaningful Euclidean metric between the data samples. If such a metric is not readily available, metric learning can be applied to construct one. Basic unsupervised techniques in this area are PCA for dimensionality reduction and the suppression of noise, or its non-linear generalisation, kernel-PCA. Supervised techniques typically work by learning linear projections that place related samples closer together, and unrelated samples farther apart. Often they are based on optimising parametric distance functions such as the Mahalanobis distance with a maximum margin criteria (Schultz & Joachims, 2003; Weinberger & Saul, 2009; Quadrianto & Lampert, 2011), or approximately minimising the leave-one-out classification error as in neighbourhood component analysis of Goldberger et al. (2004).

1.3 Infinite Latent Feature Models

Another popular approach for discovering low dimensional structure from high dimensional data is based on latent feature models. We are interested in Indian Buffet Process (IBP) based models that allow number of the latent features to be learnt from data. By defining appropriate data generating likelihood functions, the IBP can be used in, among others, binary factor analysis (Griffiths & Ghahramani, 2005), choice behaviour modelling (Görür et al., 2006), sparse factor and independent component analysis (Knowles & Ghahramani, 2007), link prediction (Miller et al., 2009), and invariant features (Austerweil & Griffiths, 2010; Zhai et al., 2012). For a recent comprehensive review of the IBP models, please refer to Griffiths & Ghahramani (2011). Lately, there is also surging interest in making the


(a) Linear Gaussian Model

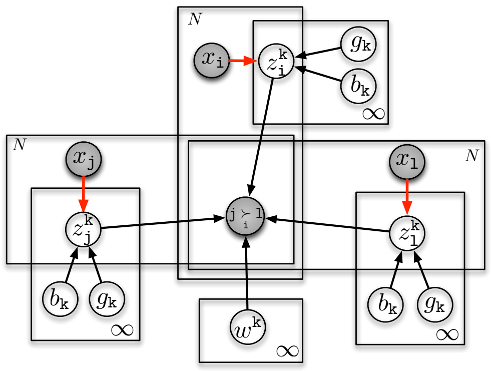

(b) Linear Probit Model

Figure 1: Graphical model for our supervised infinite latent variable models based on preference relations. Shade indicates the observed variables. The term $j \underset{i}{\succ} l = 1$ denotes that object i prefers object j to object l. 1(a): linear Gaussian model is used to generate the observed data based on latent features. 1(b): linear probit model is used to generate latent features based on the observed data. Note the difference between 1(a) and 1(b) is encoded in the direction of arrows modelling the dependency between data $\mathbf{x}$ and latent features $\mathbf{z}$.

latent representations dependent on some known degree of relatedness among observed data (Williamson et al., 2010), in learning correlated non-parametric feature models (Doshi-Velez & Ghahramani, 2009; Miller et al., 2008), or in the direction of supervised modelling, as for dimensionality reduction (Rai & Daume III, 2009). In a recent technical report, Gershman et al. (2012) express a goal closely related to ours, that nearby data is more likely to share latent features than distant data (as induced by distances between data in time or space, for example). However, encouraging sharing features between nearby data does not provide sufficient margin of separation between features of distant data. Our goal is to discover a binary latent space where meaningful notions of similarity and difference are preserved in term of metric distances.

2 The Neighbourhood Model

We are given a set of N observed data samples $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathcal{X}$, and we have used $\mathcal{X}$ to denote the input space. For example, for image objects, $\mathcal{X}$ can be features extracted based on the content of the image itself. We further assume that supervised side information is available in the form of *triplet set* $\mathcal{T} = \{(i,j,l) : \ i \sim j, \ i \not\sim l\}$, such that sample i has similar semantic concept to sample j and has no similar concept to sample l. Metaphorically, samples i and j are considered neighbours whereas samples i and l are non-neighbours. Usually this type of supervised similarity triplets do not require explicit class labels and thus are easier to obtain. For instance, in a content based image retrieval, to collect feedback, users may be required to report whether an image $\mathbf{x}_i$ looks more similar to $\mathbf{x}_j$ than it is to a third image $\mathbf{x}_l$. This task is typically much easier in comparison to labelling each individual image.

For each data point $\mathbf{x}_n$, we introduce a K dimensional vector $\mathbf{z}_n$ from a *binary* latent space, where $z_n^k = 1$ denotes that object n possesses feature k, and $z_n^k = 0$ otherwise, and K is inferred from data. Targeting directly our goal of learning neighbourhood preserving latent space that is suitable for nearest neighbour search, we require that $\mathbf{z}_i$ is similar to $\mathbf{z}_j$ to model $i \sim j$, and $\mathbf{z}_i$ is dissimilar to $\mathbf{z}_l$ to model $i \not\sim l$. The underlying idea of learning the supervised representations is based on a *folk-wisdom* principle (Goldberger et al., 2004; Weinberger & Saul, 2009; Quadrianto & Lampert, 2011) of pulling objects together if they are similar, and pushing them apart if they are not. Further, this principle is formalised as a preference relation.

When we observe that objects i and j are neighbours while objects i and l are non-neighbours, we say that object i prefers object j to object l, and use a notation $j \underset{i}{\succ} l$. Let $\mathbf{T}$ be an $N \times N \times N$ preference tensor with entries $\{t_{jl}^i\}$ where $t_{jl}^i = 1$ whenever $j \underset{i}{\succ} l$ is observed. Let $\mathbf{w}$ be a $K \times 1$ *non-negative* weight vector that affects the probability of preference relations among object i, j, and l. We assume that preference relations are independent conditioned on $\mathbf{Z}$ and $\mathbf{w}$, and furthermore only the latent features of objects i, j, and l influence the tendency of i preferring j to l. With the above assumptions, the label preference likelihood function is given by

$$\Pr(\mathbf{T}|\mathbf{Z}, \mathbf{w}) = \prod_{(i,j,l) \in \mathcal{T}} \Pr(t_{jl}^i = 1 | \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_l, \mathbf{w}). \quad (1)$$

We will subsequently use p^i_{jl} to denote $\Pr(t^i_{jl} = 1|\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_l, \mathbf{w})$. We define the individual preference probability as follows:

$$p^i_{jl} = \frac{1}{C}\sum_k w_k \mathbb{I}[z^k_i = z^k_j](1 - \mathbb{I}[z^k_i = z^k_l]), \qquad (2)$$

where $C = \sum_k w_k \mathbb{I}[z^k_i = z^k_j](1 - \mathbb{I}[z^k_i = z^k_l]) + \sum_k w_k(1-\mathbb{I}[z^k_i = z^k_j])\mathbb{I}[z^k_i = z^k_l]$ is the normalising constant. In the above, we make use of Iverson's bracket notation: $\mathbb{I}[P] = 1$ for the condition P is true and it is 0 otherwise. The term $\sum_k w_k \mathbb{I}[z^k_i = z^k_j](1-\mathbb{I}[z^k_i = z^k_l])$ collects the weights for all features that object i and j have but object l does not have, and the weights for all features that object i and j do not have, but l has. Thus, the choice between two alternatives j and l from point-of-view i depends on latent features that are shared between i and j but not l. This type of preference model (2) is inspired by the choice model of Görür et al. (2006) and is based on a standard Restle's choice model in psychology (Restle, 1961).

We take a fully Bayesian approach by treating latent variables $\mathbf{Z}$ and $\mathbf{w}$, as random variables, and computing the posterior distribution over them by invoking Bayes' theorem. We discuss the selection of prior probabilities on $\mathbf{Z}$ and $\mathbf{w}$ in detail in the next section.

3 The Generative Process

We want to define a flexible prior on $\mathbf{Z}$ that allows simultaneous inference of the number of features and all the entries in $\mathbf{Z}$ at the same time. We will thus put the Indian Buffet Process (IBP) prior (Griffiths & Ghahramani, 2005) on $\mathbf{Z}$. The IBP is a prior on infinite binary matrices such that with probability one, a feature matrix drawn from it will only have a finite number of non-zero features for a finite number of samples (entries). More importantly, the IBP prior has a full support for any feature matrix regardless of the number of non-zero features it has. We choose to put a *Gamma* distribution as a prior for the elements of $\mathbf{w}$. This is a natural prior for a non-negative weight vector. Section 2 describes a likelihood function for modelling the supervised side information, the next required modelling is to define the data likelihood. We explore two directions: one is to use a standard linear Gaussian model which assumes data are generated via a linear superposition of latent features. The second one is to make the latent features be dependent on observed data via a novel and simple linear probit model. We discuss both models in the next sections (refer to Figure 1 for graphical model representations).

3.1 $\mathbf{Z} \rightarrow \mathbf{X}$ Linear Gaussian Feature Model

This data generating model was initially explored for the IBP in the *unsupervised* context by Griffiths & Ghahramani (2005). In this model, for an M-dimensional input space $\mathcal{X} = \mathbb{R}^M$, the data point $\mathbf{x}_n \in \mathbb{R}^M$ is generated as follows:

$$\mathbf{x}_n = \mathbf{V}\mathbf{z}_n + \sigma_x \boldsymbol{\epsilon} \text{ where } \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|\mathbf{0}, \mathbf{I}). \qquad (3)$$

In the above, $\mathbf{V}$ is a real-valued $M \times K$ matrix of weights. We use a spherical Gaussian conjugate prior with a covariance matrix $\sigma^2_v\mathbf{I}$ for this feature weight matrix, $\mathbf{V}$. The generative process for our preference model with a linear Gaussian likelihood is then:

$$\mathbf{Z} \sim \text{IBP}(\alpha);\ \mathbf{V} \sim \mathcal{N}(\mathbf{0}, \sigma^2_v\mathbf{I});\ \mathbf{x}_n|\mathbf{z}_n, \mathbf{V} \sim \mathcal{N}(\mathbf{V}\mathbf{z}_n, \sigma^2_x\mathbf{I});$$
$$w_k \overset{\text{i.i.d.}}{\sim} \mathcal{G}(\gamma_w, \theta_w);\ j \underset{i}{\succ} l|\mathbf{Z}, \mathbf{w} \sim \text{Bernoulli}(p^i_{jl}),$$

where p^i_{jl} is defined in Equation (2). We can subsequently compute the posterior distribution of the latent feature matrix $\mathbf{Z}$ and the weights $\mathbf{w}$ using the conditional independence assumptions depicted in Figure 1(a). This is given as, $\Pr(\mathbf{Z}, \mathbf{w}|\mathbf{X}, \mathbf{T}) \propto$

$$\int \Pr(\mathbf{T}|\mathbf{Z}, \mathbf{w})\Pr(\mathbf{X}|\mathbf{Z}, \mathbf{V}, \sigma_x)\Pr(\mathbf{Z}|\alpha)\Pr(\mathbf{V}|\sigma_v)\Pr(\mathbf{w}|\gamma_w, \theta_w)d\mathbf{V}. \qquad (4)$$

3.2 $\mathbf{X} \rightarrow \mathbf{Z}$ Linear Probit Dependent Model

The neighbourhood model with linear Gaussian data likelihood requires the inferred latent features to *explain* supervised similarity in given triplets and to *generate* observed data. Modelling observed data is a hard task by itself. Instead, we can devote the latent features to model supervised similarity triplets and to have an IBP model where the probability of a feature k being on is *dependent* on some object covariate information $\mathbf{x}_n \in \mathbb{R}^M$. To achieve a dependent IBP model, we start with the stick breaking construction of the IBP (Teh et al., 2007):

$$z^k_n|b_k \sim \text{Bernoulli}(b_k);\quad b_k := v_k b_{k-1} = \prod_{j=1}^{k} v_j \qquad (5)$$
$$v_j \sim \text{Beta}(\alpha, 1) \text{ and } b_0 = 1. \qquad (6)$$

Williamson et al. (2010) observe that a Bernoulli(β) random variable z can be represented as

$$z = \mathbb{I}[u < \Phi^{-1}(\beta|\mu, \sigma^2)] \qquad (7)$$
$$u \sim \mathcal{N}(\mu, \sigma^2), \qquad (8)$$

where $\Phi(\cdot|\mu, \sigma^2)$ is a Gaussian cumulative distribution function (CDF), and for simplicity we focus on the standard Gaussian CDF, that is $\Phi_{0,1}(\cdot) := \Phi(\cdot|0, 1)$.

Therefore, we propose a simple alternative to dependent model of Williamson et al. (2010) by linearly parameterising the cut off variable u_n^k, as follows:

$$z_n^k = \mathbb{I}[u_n^k < \Phi_{0,1}^{-1}(b_k)] \tag{9}$$
$$u_n^k = -\mathbf{x}_n^\top \mathbf{g}_k + \epsilon, \tag{10}$$

where $\mathbf{g}_k \in \mathbb{R}^M$ is a vector of regression coefficients for each feature k, and $\epsilon \sim \mathcal{N}(0,1)$. Equivalently we can integrate out ϵ, and write $\Pr(z_n^k = 1|\mathbf{x}_n, \mathbf{g}_k, b_k)$

$$= \int \mathbb{I}[\epsilon < \mathbf{x}_n^\top \mathbf{g}_k + \Phi_{0,1}^{-1}(b_k)]\mathcal{N}(\epsilon)d\epsilon \tag{11}$$
$$= \Phi_{0,1}(\mathbf{x}_n^\top \mathbf{g}_k + \Phi_{0,1}^{-1}(b_k)). \tag{12}$$

The interpretation of the dependent model above is, that whether a feature k is on is given by a *probit regression* model, with decreasing biases $\Phi_{0,1}^{-1}(b_k)$, which will ensure that only finitely many features are used. Note that this scenario of dependence on per object covariates $\mathbf{x}_n$ is not covered by the dependent IBP of Williamson et al. (2010). Their model defines a prior over multiple IBP matrices which (for certain settings of the model) are marginally IBP: a similar statement for our construction is meaningless since we only have one IBP matrix. However, our model does have the property that $\mathbf{Z}$ is IBP distributed conditional on $\mathbf{g}_k = 0$ for all k. We use a spherical Gaussian conjugate prior with a covariance matrix $\sigma_g^2\mathbf{I}$ for the regression coefficient matrix, $[\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_K] := \mathbf{G}$. With the above construction, the generative process for our preference model with linear probit likelihood is then:

$$v_j \sim \text{Beta}(\alpha, 1); \quad b_k = \prod_{j=1}^{k} v_j; \quad \mathbf{G} \sim \mathcal{N}(\mathbf{0}, \sigma_g^2\mathbf{I});$$
$$z_n^k|\mathbf{x}, \mathbf{g}, \mathbf{b} \sim \text{Bernoulli}(\Phi_{0,1}(\mathbf{x}_n^\top \mathbf{g}_k + \Phi_{0,1}^{-1}(b_k)));$$
$$w_k \overset{\text{i.i.d.}}{\sim} \mathcal{G}(\gamma_w, \theta_w); \; j \underset{i}{\succ} l|\mathbf{Z}, \mathbf{w} \sim \text{Bernoulli}(p_{jl}^i),$$

The posterior distribution of the latent feature matrix $\mathbf{Z}$, the feature presence probability $\mathbf{b}$, the weights $\mathbf{w}$, and the regression coefficient matrix $\mathbf{G}$ using the conditional independence assumptions depicted in Figure 1(b) is then $\Pr(\mathbf{Z}, \mathbf{b}, \mathbf{w}, \mathbf{G}|\mathbf{X}, \mathbf{T}) \propto$

$$\Pr(\mathbf{T}|\mathbf{Z}, \mathbf{w})\Pr(\mathbf{Z}|\mathbf{X}, \mathbf{G}, \mathbf{b})\Pr(\mathbf{w}|\gamma_w, \theta_w)\Pr(\mathbf{G}|\sigma_g)\Pr(\mathbf{b}|\alpha). \tag{13}$$

4 Inference

In the inference phase, the goal is to compute the joint posterior over the latent binary feature matrix $\mathbf{Z}$, the non-negative weights $\mathbf{w}$ and the regression coefficient matrix $\mathbf{G}$ (for the linear probit dependent model) as expressed in (4) and (13). For our proposed model, exact inference is computationally intractable. Thus, we employ a Markov Chain Monte Carlo (MCMC) method to explore the posterior distributions.

4.1 Sampling for Linear Gaussian Model

M-H sampling of Z: The sampler for the binary feature matrix $\mathbf{Z}$ consists of sampling existing features, proposing new features with corresponding weights, and accepting or rejecting them based on the Metropolis-Hasting (M-H) criterion. We sample each row $\mathbf{z}_n$ one after another. For sampling existing features, we have: $\Pr(z_n^k = 1|\mathbf{X}, \mathbf{T}, \mathbf{w}) \propto$

$$\int m_{-n,k}\Pr(\mathbf{T}|\mathbf{w}, Z_{-n,k}, z_n^k = 1)\Pr(\mathbf{X}|Z_{-n,k}, z_n^k = 1, \mathbf{V})\Pr(\mathbf{V})d\mathbf{V}, \tag{14}$$

where $m_{-n,k}$ denote the number of non-zero entries in column k excluding row n. For sampling new features, we simultaneously propose $(K_{\text{new}}, \mathbf{Z}_{\text{new}}, \mathbf{w}_{\text{new}})$ where a number K_{new} for new features are sampled from the prior Poisson(α/N). We propose $\mathbf{w}_{\text{new}}$ from its Gamma prior. We consider this proposal with a M-H acceptance ratio which reduces to the ratio of the likelihoods (Meeds et al., 2007).

Slice sampling of w: We sample each of the non-negative weights that correspond to the non-zero features and drop the weights that correspond to zero features. We use a slice sampling procedure of Neal (2003). Due to our Gaussian assumptions, the real-valued weight matrix $\mathbf{V}$ in (14) can be marginalised analytically (Griffiths & Ghahramani, 2005).

4.2 Sampling for Linear Probit Model

We adapt a slice sampling procedure with stick breaking representation of Teh et al. (2007).

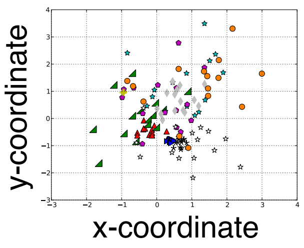
Adaptive rejection sampling of b: The form of the conditional distribution of $\mathbf{b}$ can be found in Teh et al. (2007), and due to log-concavity of this distribution, Teh et al. (2007) suggest to use adaptive rejection sampling (ARS) (Gilks & Wild, 1992) to draw samples.

Sampling of Z: As in Teh et al. (2007), given the auxiliary slice variable, we will only update the latent feature for each observation and each dimension where its feature presence probability is below the slice. The required conditional distributions for our case are: $\Pr(z_n^k = 1|\mathbf{x}_n, \mathbf{T}, \mathbf{w}, \mathbf{g}_k, b_k) \propto$

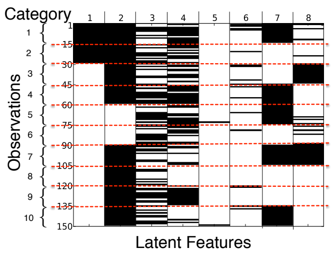
$$\Phi_{0,1}(\mathbf{x}_n^\top \mathbf{g}_k + \Phi_{0,1}^{-1}(b_k))\Pr(\mathbf{T}|\mathbf{w}, Z_{-n,k}, z_n^k = 1). \tag{15}$$

Slice sampling of w: Similar to the linear Gaussian case, we update the non-negative weights using a slice sampling procedure.

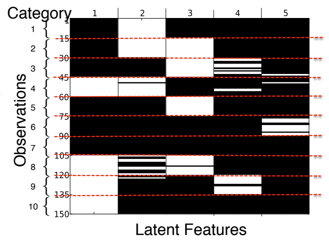
Elliptical slice sampling of G: We propose to sample each component $\mathbf{g}_k$ of the regression coefficient matrix $[\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_K]$ using elliptical slice sampling (ESS) (Murray et al., 2010), an efficient MCMC procedure for training of tightly coupled latent variables with a Gaussian prior.



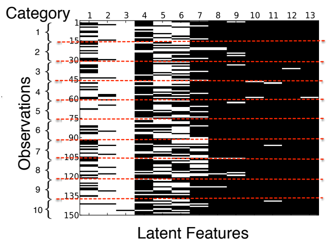
(a) Synthetic Data



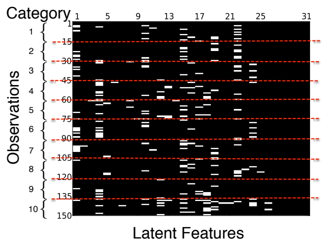
(b) Our Super Gaussian IBP



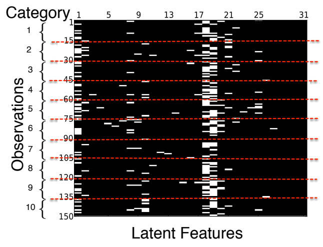
(c) Our Super Probit IBP



(d) IBP



(e) Input dd-IBP



(f) Output dd-IBP

Figure 2: Visualisation of the binary latent space. 2(a): 150 synthetic data points with 10 categories from a mixture of 2-D multivariate Gaussians. 2(b)-2(f): The corresponding binary representations of the data generated by various methods ('1' is white, and '0' is black). The supervised side information is given by a set of triplets. In this example, each training data point has L neighbours from the same category and L non-neighbours from different categories encoded in the form of triplets (*sample*, its *neighbour*, its *non-neighbour*). Here, the binary representations are visualised by grouping observations according to their categories. Our methods preserve given triplets structure by assigning distinct features for different categories.

5 Prediction on Test Data

5.1 Linear Gaussian Model

For a *previously unseen* test point $\mathbf{x}_* \in \mathbb{R}^M$, the joint predictive distribution for the latent variable $\mathbf{z}_*$ and the preference relation variable t_* is: $\Pr(\mathbf{z}_*, t_* | \mathbf{X}, \mathbf{T}, \mathbf{x}_*) =$

$$\int \sum_{\mathbf{Z}} \Pr(\mathbf{z}_*, t_* | \mathbf{Z}, \mathbf{w}, \mathbf{X}, \mathbf{x}_*) \Pr(\mathbf{Z}, \mathbf{w} | \mathbf{X}, \mathbf{T}) d\mathbf{w}, \text{ where}$$

$\Pr(\mathbf{z}_*, t_* | \mathbf{Z}, \mathbf{w}, \mathbf{X}, \mathbf{x}_*) = \Pr(t_* | \mathbf{z}_*, \mathbf{w}) \Pr(\mathbf{z}_* | \mathbf{Z}, \mathbf{X}, \mathbf{x}_*)$. This involves averaging over the predictions made by each of the posterior samples of $\mathbf{Z}$ and $\mathbf{w}$. The preference relation variable t_* is a binary variable representing whether object $\mathbf{x}_*$ is preferred or not in some triplet. Since we have trained the binary latent space in a supervised manner, we could predict the neighbours and non-neighbours of the new test point by performing a *nearest neighbour* classification of the inferred test latent variable $\mathbf{z}_*$ with respect to the training latent variables $\mathbf{Z}$. Therefore, we are interested only in the predictive distribution over the latent variable $\mathbf{z}_*$, and it is in the form of:

$$\Pr(\mathbf{z}_* | \mathbf{X}, \mathbf{x}_*) = \sum_{\mathbf{Z}} \Pr(\mathbf{z}_* | \mathbf{Z}, \mathbf{X}, \mathbf{x}_*) \Pr(\mathbf{Z} | \mathbf{X}), \text{ where}$$

$$\Pr(\mathbf{z}_* | \mathbf{Z}, \mathbf{X}, \mathbf{x}_*) \propto \Pr(\mathbf{x}_* | \mathbf{z}_*, \mathbf{Z}, \mathbf{X}) \Pr(\mathbf{z}_* | \mathbf{Z}). \tag{16}$$

We notice that the explicit form of $\Pr(\mathbf{x}_* | \mathbf{z}_*, \mathbf{Z}, \mathbf{X})$ is

$$\begin{bmatrix} \mathbf{X} \\ \mathbf{x}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{Z}\mathbf{Z}^\top + \sigma_x^2/\sigma_v^2 \mathbf{I} & \mathbf{Z}\mathbf{z}_*{}^\top \\ \mathbf{z}_*\mathbf{Z}^\top & \mathbf{z}_*\mathbf{z}_*{}^\top + \sigma_x^2/\sigma_v^2 \mathbf{I} \end{bmatrix}\right),$$

thus $\Pr(\mathbf{x}_* | \mathbf{z}_*, \mathbf{Z}, \mathbf{X}) \sim \mathcal{N}(\mu_*, \mathbf{\Sigma}_*)$ where

$$\mu_* = \mathbf{z}_* (\mathbf{Z}^\top \mathbf{Z} + \sigma_x^2/\sigma_v^2 \mathbf{I})^{-1} \mathbf{Z}^\top \mathbf{X} \tag{17a}$$

$$\mathbf{\Sigma}_* = \mathbf{z}_* \mathbf{z}_*{}^\top - \mathbf{z}_* (\mathbf{Z}^\top \mathbf{Z} + \sigma_x^2/\sigma_v^2 \mathbf{I})^{-1} \mathbf{Z}^\top \mathbf{Z} \mathbf{z}_*{}^\top. \tag{17b}$$

The above predictive distribution $\Pr(\mathbf{x}_* | \mathbf{z}_*, \mathbf{Z}, \mathbf{X})$ defines a distribution of the mapping from a latent space to the observed data space.

Fast approximation In cases where we are only interested in a maximum a posteriori (MAP) estimate of the latent variables, it is desirable to avoid sampling from the predictive distribution, and directly find an approximate MAP estimate in a computationally efficient way. In our case, we use the predictive mean of $\Pr(\mathbf{x}_* | \mathbf{z}_*, \mathbf{Z}, \mathbf{X})$ in (17a) to approximate $\mathbf{z}_*$ by solving a linear system of equations, resulting in a continuous estimate $\hat{\mathbf{z}}_*$ of the binary vector $\mathbf{z}_*$.

5.2 Linear Probit Dependent Model

Similar to the linear Gaussian model but with explicit representation of the regression coefficient matrix, the joint predictive distribution of the latent variable $\mathbf{z}_*$ and the preference variable t_* for a *new* test point $\mathbf{x}_* \in \mathbb{R}^M$ is: $\Pr(\mathbf{z}_*, t_* | \mathbf{T}, \mathbf{x}_*) =$

$$\iiint \Pr(\mathbf{z}_*, t_* | \mathbf{G}, \mathbf{w}, \mathbf{b}, \mathbf{x}_*) \sum_{\mathbf{Z}} \Pr(\mathbf{Z}, \mathbf{b}, \mathbf{w}, \mathbf{G} | \mathbf{X}, \mathbf{T}) d\mathbf{b} d\mathbf{w} d\mathbf{G},$$

with test likelihood given as follows:

$$\Pr(\mathbf{z}_*, t_* | \mathbf{G}, \mathbf{w}, \mathbf{b}, \mathbf{x}_*) = \Pr(t_* | \mathbf{z}_*, \mathbf{w}) \Pr(\mathbf{z}_* | \mathbf{G}, \mathbf{b}, \mathbf{x}_*). \tag{18}$$

As earlier, we are only concerned with the predictive distribution over the latent variable $\mathbf{z}_*$ for the new input $\mathbf{x}_*$, that is $\Pr(\mathbf{z}_* | \mathbf{G}, \mathbf{b}, \mathbf{x}_*)$. Based on our linear probit model, this will simply be $\Pr(\mathbf{z}_*^k = 1 | \mathbf{G}, b_k, \mathbf{x}_*) = \Phi_{0,1}(\mathbf{x}_*^\top \mathbf{g}_k + \Phi_{0,1}^{-1}(b_k))$.

6 Experiments

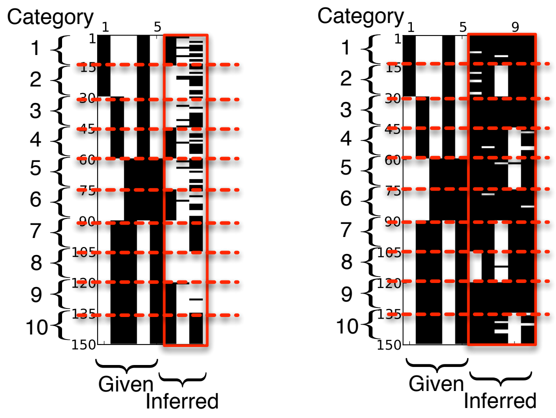
To assess the efficacy of our models, we perform two sets of experiments. We start with a synthetic data experiment to explore the structure of the latent space $\mathbf{Z}$ produced by the proposed models (Section 6.1-6.2). Our second experiment is extending hash codes in image data (Section 6.3).

6.1 Visualisation of the Binary Latent Spaces

Data We generate 150 synthetic data points with 10 categories from a mixture of 2-D multivariate Gaussians with uniformly drawn standard deviations in the range $[0, 1]$. The means are uniformly drawn in the range $[-1, 1]$ *per category*. The visualisation of the generated data is provided at Figure 2(a).

Algorithms We compare the generated latent space of our supervised linear Gaussian (`Super Gaussian IBP`) and supervised linear probit (`Super Probit IBP`) models with the Indian buffet process (`IBP`), and the distance dependent Indian buffet process with distance defined on $\mathcal{X}$ (`Input dd-IBP`), and on the labels (`Output dd-IBP`)[1]. The supervised side information is given by a set of triplets generated the same way as in Weinberger & Saul (2009). Specifically, for each training data point, we are given its L neighbours from the same category, and L non-neighbours from different category encoded in the form of triplets (in this experiment we use $L = 15$). As a practical note, triplets as supervised side information correspond only to a small number of observed entries

[1]We use the implementation provided by Gershman et al. (2012) at `http://www.princeton.edu/~sjgershm/ddIBP_release.zip`



(a) `Super Gaussian IBP` (b) `Super Probit IBP`

Figure 3: Extending observed binary variables. The first 5 *given* binary variables do not respect the neighbourhood structure, for example, categories 1 and 2 have the same '01101' representation. Our supervised models allow *coupling* between given and inferred latent features. As a result, the inferred latent features enforce separation among categories and amend shortcomings that the observed binary representations might have in respecting the neighbourhood structure.

in $\mathbf{T}$. This translates to a small computational overhead compared to the standard IBP inference. For all methods, we place conjugate priors on the hyperparameters, and subsequently perform posterior inference over them. Our `Super Gaussian IBP` and `Super Probit IBP` methods discover binary representations which preserve neighbourhood structure. The results are shown in Figure 2. In comparison to the IBP 2(d), dd-IBP input 2(e), and dd-IBP output 2(f) models, the inferred feature matrix $\mathbf{Z}$ of the proposed models 2(b) and 2(c) appears to have a notable supervised structure for all categories by assigning distinct features for different categories.

6.2 Model with *Observed* Binary Variables

We are interested to explore how our model can be used to extend an existing binary hash of the data. To do so, we assume that we are already given 5 binary variables that partially separates the objects according to their categories. In this case, categories 1 and 2 have binary representations '01101', categories 3 and 4 have '10101' representations, categories 5 and 6 have '11000', and categories 7, 8, 9 and 10 have 10010. Refer to the first 5 dimensions of Figure 3(a) for illustration. The task is to extend the binary vector to model the supervised neighbourhood information, thereby disambiguating classes with the same observed

Table 1: Extending hash codes results for image data. k-NN accuracy mean ± std over 5 random repeats. `IBP`: standard IBP algorithm (Griffiths & Ghahramani, 2005); `dd-IBP`: distance dependent IBP (Gershman et al., 2012) where `Input`: distance on $\mathcal{X}$, and `Output`: distance on the labels; `Super Gaussian IBP`: Our proposed supervised IBP with linear Gaussian feature model; `Super Probit IBP`: Our proposed supervised IBP with linear probit dependent model; `Reference`: original 128 real-valued feature representations. The best result and those not significantly worse than it, are in **boldface**. We use a one-sided paired t-test with 95% confidence.

	Hash	IBP	Input dd-IBP	Output dd-IBP	Super Gaussian IBP	Super Probit IBP	Reference
				5 animal categories			
1 NN	26.3±2.2	30.3±2.0	27.9±6.3	29.7±3.5	33.8±1.6	**42.8±2.4**	**40.9±4.7**
3 NN		29.5±2.9	29.6±3.6	31.0±1.4	34.6±2.3	**41.9±3.4**	**40.2±3.4**
15 NN		31.5±2.6	27.8±2.8	28.1±3.2	35.5±1.0	**44.5±2.1**	39.3±3.7
30 NN		29.5±3.2	24.3±3.0	23.6±3.4	33.8±0.7	**45.9±4.1**	36.1±2.8
				10 animal categories			
1 NN	12.7±2.5	17.1±3.1	12.9±2.6	15.9±2.3	17.3±1.2	**25.0±2.9**	**25.0±2.2**
3 NN		17.9±2.8	13.1±2.4	15.3±2.3	18.2±1.2	**25.1±3.0**	**26.0±1.7**
15 NN		16.4±2.5	14.7±2.3	15.1±1.8	18.0±1.5	**26.6±2.7**	**27.8±2.8**
30 NN		17.7±3.4	14.5±1.9	14.0±1.9	18.3±1.4	**27.5±2.4**	**25.8±1.4**

binary hash. In this setting, we want to extend the observed binary representations $\mathbf{h_n} \in \mathcal{H}$ (for each example $\mathbf{x_n}$) where $\mathcal{H} \in \{0,1\}^D$ with a *latent* binary feature $\mathbf{z_n}$, forming an extended representation $[\mathbf{h_n}^\top \mathbf{z_n}^\top]^\top$. Let $\mathbf{H}$ be the observed $N \times D$ binary representation matrix (D = 5 in our experiment) and $\mathbf{w_H}$ be a $D \times 1$ *non-negative* weight vector. The full preference probability is now, $p^i_{jl} = \frac{1}{C}\sum_k w_k \mathbb{I}[z^k_i = z^k_j](1 - \mathbb{I}[z^k_i = z^k_l]) + \sum_d w^d_H \mathbb{I}[h^d_i = h^d_j](1 - \mathbb{I}[h^d_i = h^d_l])$, where the normalising constant C ensures $p^i_{jl} + p^i_{lj} = 1$. The latent features are inferred to enforce separation among categories and amend shortcomings that the observed binary variables might have in respecting the neighbourhood structure.

Results The supervised models are trained to utilise the *given* binary features, and to add additional binary latent representations only when it is needed to support the discrimination between categories (see Figure 3). As an example, in case of categories 1 and 2 that are indistinguishable under the given 5 binary vectors, 3(a)-3(b) learn at least unit distance in the extended representation for these categories, and increase the separation of the codes for the rest of categories. For this example, `Super Gaussian IBP` (Figure 3(a)) discovers additional 3 binary latent variables where category 1 has '0∗∗' and category 2 has '1∗∗'. While `Super Probit IBP` (Figure 3(b)) discovers 5 more binary latent variables with '$**0**$' and '$**1**$' assigned to category 1 and 2, respectively.

6.3 Extending Hash Codes Application

In this experiment, we assume that we are given binary hash codes, for example, via a spectral hashing method (Weiss et al., 2009) or via binary attribute predictors (Lampert et al., 2009) explained in the subsequent section, and our goal is to extend these observed codes with latent binary features. We expect the *extended codes* to have a better nearest neighbour search performance, especially in the case where the hash codes do not respect the neighbourhood structure of data.

Data We use the Animals with Attributes dataset [2]. The dataset consists of 30,475 images. Each of the images has a category label which corresponds to the animal class. There are 50 animal classes in this dataset. The dataset also contains semantic information in the form of an 85-dimensional Osherson's (Osherson et al., 1991; Kemp et al., 2006) attribute vector for each animal class describing colour, texture, shape, among others. Images are represented by colour histograms of quantised RGB pixels with a codebook of size 128.

Hash Codes The hash codes at training phase are given by the Osherson's attribute vector. In the testing phase, we build hash codes using attribute predictors trained offline (Lampert et al., 2009). We generate the hash codes as follows: each class is assigned an attribute binary string of length D (in our case, the Osherson's vector), subsequently we learn D logistic regression functions, one for each bit position in these binary strings. When a new data point arrives, we evaluate each of the D *test* logistic regressors to generate a D-bit hash code.

Results We use 27,032 images from 45 classes to be our *initial image corpus* for learning the attribute hash codes. We use the colour histograms to represent images, and we focus on colour attributes hash codes, which corresponds to the first 5-bits in the Osherson's attribute vector. This simulates the case where a large pool of data is available for building the hash codes. From the remaining five classes, we randomly sample

[2]http://attributes.kyb.tuebingen.mpg.de/

Table 2: Accuracy comparison between SVM with a linear kernel and Super Probit IBP. The best result and those not significantly worse than it, are in **boldface** (one-sided paired t-test with 95% confidence).

	5 categories	**10 categories**
`Linear SVM`	**43.3±4.0**	**29.0±3.3**
`Super Probit IBP`	**45.9±4.1 (30 NN)**	**27.5±2.4 (30 NN)**

300 images with uniform class proportions to form a *refinement set* for training our models, and the test set using 50/50 split. Our refinement set simulates the case where training samples are from different categories than in initial corpus, and therefore have different unseen properties. We repeat the above procedure for a refinement set with 10 new categories. That is, we use $23,266$ images from 40 classes to learn the hash codes, and randomly sample 600 images from the remaining 10 classes for training and test with 50/50 split. The supervised similarity triplets are formed in the same way as in synthetic experiments ($L = 30$). We note that the costly MCMC procedure is performed offline at the training phase. At test time, we simply perform a fast approximation via matrix vector multiplication in linear Gaussian (Section 5.1) or compute probit regression in linear probit (Section 5.2).

The full results with a comparison to the predicted hash codes using logistic regressors and the standard `IBP` and `dd-IBP` are summarised in Table 1[3]. We observe that our proposed models, `Super Gaussian IBP` and `Super Probit IBP`, exceed the performance of IBP and dd-IBP in *all cases*. We further notice that Super Probit IBP is far superior to Super Gaussian IBP. We credit this to the fact that linear Gaussian models are less suitable for modelling real-valued images (Austerweil & Griffiths, 2010; Zhai et al., 2012). One of the solutions would be to define a more sophisticated likelihood function (Austerweil & Griffiths, 2010; Zhai et al., 2012). Instead in this work we focus on generating binary features that depend on the observed images via probit regression. As a reference, we also provide k-NN performance in the original 128 real-valued features. Original features will require storage of $8,192$ ($128*64$) bits per image, while our Super Probit IBP code with 80 inferred binary latent dimensions will only consume approximately 80 bits per image and gives better results. Further, to put our results in a wider perspective, we also provide results of running SVM[4] on the original real-valued features with a linear kernel in Table 2.

[3]The k-NN performance of the hash method does not depend on k, because for training we use the *given* Osherson's colour hash codes defined per class.

[4]We use the LIBSVM library available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

Table 3: Effect of Bayesian Averaging on Super Probit IBP. Accuracy mean±std. `last`: using a sample from the last iteration; `average`: using samples from the last 50 iterations. **boldface** is significant using a one-sided paired t-test with 95% confidence.

	5 animal categories		**10 animal categories**	
	`last`	`average`	`last`	`average`
1 NN	42.8±2.4	42.7±3.2	25.0±2.9	25.5±3.9
3 NN	41.9±3.4	**44.0±2.7**	25.1±3.0	**27.1±2.9**
15 NN	44.5±2.1	**46.5±2.3**	26.6±2.7	27.7±2.3
30 NN	45.9±4.1	46.5±2.3	27.5±2.4	27.2±2.7

Bayesian approach allows us to learn a distribution over hash codes. In our experiments, we run MCMC until a fixed number of iterations, and subsequently consider the hash codes given by the last iteration. We can instead exploit the full distribution by averaging the nearest neighbour retrieval performances after burn-in period. The results of Bayesian averaging on Super Probit IBP are summarised in Table 3. Clearly, averaging has a *positive effect* in the performance, however, with a price in storage requirement where now several databases have to be maintained.

7 Discussion and Conclusion

We have presented probabilistic models to simultaneously infer the number of binary latent variables, and their values so as to preserve a given neighbourhood structure. The models map objects in the same semantic concept to similar latent values, and objects in different concepts to dissimilar latent values. We substantiate our claim that the proposed supervised models encourage coupling among latent features by showing that when given binary representations, the models utilise the given representation, and add dimensions in a latent space when it is needed to preserve the neighbourhood structure.

Our experiments in a nearest neighbour search show that our methods are able to find semantically similar neighbours due to the supervised nature of the latent space, and far exceed the performance of other state-of-the-art infinite latent variable models, such as the standard Indian buffet process (IBP) and its recent extension, the distance dependent IBP (dd-IBP).

Acknowledgements

The authors would like to thank Kurt Miller, Dilan Görür, and Christoph Lampert for discussions. NQ is supported by a Newton International Fellowship. The work is done while VS and DAK are at University of Cambridge.

References

Arya, Sunil, Mount, David M., Netanyahu, Nathan S., Silverman, Ruth, and Wu, Angela Y. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 1998.

Austerweil, Joseph and Griffiths, Tom. Learning invariant features using the transformed indian buffet process. In *NIPS*, 2010.

Beygelzimer, Alina, Kakade, Sham, and Langford, John. Cover trees for nearest neighbor. In *ICML*, 2006.

Doshi-Velez, Finale and Ghahramani, Zoubin. Correlated non-parametric latent feature models. In *UAI*, 2009.

Gershman, Samuel J., Frazier, Peter I., and Blei, David M. Distance dependent infinite latent feature models. Technical report, arXiv, 2012.

Gilks, Walter R. and Wild, Pascal. Adaptive rejection sampling for gibbs sampling. *Applied Statistics*, 1992.

Goldberger, Jacob, Roweis, Sam T., Hinton, Geoffrey E., and Salakhutdinov, Ruslan. Neighbourhood components analysis. In *NIPS*, 2004.

Görür, Dilan, Jäkel, Frank, and Rasmussen, Carl Edward. A choice model with infinitely many latent features. In *ICML*, 2006.

Griffiths, Thomas L. and Ghahramani, Zoubin. Infinite latent feature models and the indian buffet process. In *NIPS*, 2005.

Griffiths, Thomas L. and Ghahramani, Zoubin. The indian buffet process: An introduction and review. *Journal of Machine Learning Research*, 2011.

Indyk, Piotr and Motwani, Rajeev. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC*, 1998.

Kemp, Charles, Tenenbaum, Joshua B., Griffiths, Thomas L., Yamada, Takeshi, and Ueda, Naonori. Learning systems of concepts with an infinite relational model. In *AAAI*, 2006.

Knowles, David and Ghahramani, Zoubin. Infinite sparse factor analysis and infinite independent components analysis. In *ICA*, 2007.

Lampert, Christoph H., Nickisch, Hannes, and Harmeling, Stefan. Learning to detect unseen object classes by betweenclass attribute transfer. In *CVPR*, 2009.

Meeds, Edward, Ghahramani, Zoubin, Neal, Radford M., and Roweis, Sam T. Modeling dyadic data with binary latent factors. In *NIPS*, 2007.

Miller, Kurt, Griffiths, Thomas, and Jordan, Michael. Nonparametric latent feature models for link prediction. In *NIPS*, 2009.

Miller, Kurt T., Griffiths, Thomas L., and Jordan, Michael I. The phylogenetic indian buffet process: A non-exchangeable nonparametric prior for latent features. In *UAI*, 2008.

Mu, Yadong, Shen, Jialie, and Yan, Shuicheng. Weakly-supervised hashing in kernel space. In *CVPR*, 2010.

Murray, Iain, Adams, Ryan Prescott, and MacKay, David J. C. Elliptical slice sampling. *AISTATS*, 2010.

Neal, Radford M. Slice sampling. *Annals of Statistics*, 2003.

Norouzi, Mohammad, Fleet, David J., and Salakhutdinov, Ruslan. Hamming distance metric learning. In *NIPS*, 2012.

Osherson, Daniel N., Stern, Joshua, Wilkie, Ormond, Stob, Michael, and Smith, Edward E. Default probability. *Cognitive Science*, 1991.

Quadrianto, Novi and Lampert, Christoph H. Learning multi-view neighborhood preserving projections. In *ICML*, 2011.

Rai, Piyush and Daume III, Hal. Multi-label prediction via sparse infinite CCA. In *NIPS*, 2009.

Restle, Frank. *Psychology of judgment and choice: A theoretical essay.* John Wiley & Sons, 1961.

Salakhutdinov, Ruslan and Hinton, Geoffrey E. Semantic Hashing. In *SIGIR workshop on Information Retrieval and applications of Graphical Models*, 2007.

Schultz, Matthew and Joachims, Thorsten. Learning a distance metric from relative comparisons. In *NIPS*, 2003.

Teh, Yee Whye, Görür, Dilan, and Ghahramani, Zoubin. Stick-breaking construction for the indian buffet process. *AISTATS*, 2007.

Torralba, Antonio B., Fergus, Robert, and Weiss, Yair. Small codes and large image databases for recognition. In *CVPR*, 2008.

Wang, Jun, Kumar, Sanjiv, and Chang, Shih-Fu. Semi-supervised hashing for large-scale search. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2012.

Weinberger, Kilian Q. and Saul, Lawrence K. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 2009.

Weiss, Yair, Torralba, Antonio, and Fergus, Rob. Spectral hashing. In *NIPS*, 2009.

Williamson, Sinead, Orbanz, Peter, and Ghahramani, Zoubin. Dependent indian buffet processes. *AISTATS*, 2010.

Zhai, Ke, Hu, Yuening, Boyd-Graber, Jordan L., and Williamson, Sinead. Modeling images using transformed indian buffet processes. In *ICML*, 2012.

Normalized online learning

Stéphane Ross
Carnegie Mellon University
Pittsburgh, PA, USA
stephaneross@cmu.edu

Paul Mineiro
Microsoft
Bellevue, WA, USA
paul.mineiro@gmail.com

John Langford
Microsoft Research
New York, NY, USA
jcl@microsoft.com

Abstract

We introduce online learning algorithms which are independent of feature scales, proving regret bounds dependent on the ratio of scales existent in the data rather than the absolute scale. This has several useful effects: there is no need to pre-normalize data, the test-time and test-space complexity are reduced, and the algorithms are more robust.

1 Introduction

Any learning algorithm can be made invariant by initially transforming all data to a preferred coordinate system. In practice many algorithms begin by applying an affine transform to features so they are zero mean with standard deviation 1 [Li and Zhang, 1998]. For large data sets in the batch setting this preprocessing can be expensive, and in the online setting the analogous operation is unclear. Furthermore preprocessing is not applicable if the inputs to the algorithm are generated dynamically during learning, e.g., from an on-demand primal representation of a kernel [Sonnenburg and Franc, 2010], virtual example generated to enforce an invariant [Loosli et al., 2007], or machine learning reduction [Allwein et al., 2001].

When normalization techniques are too expensive or impossible we can either accept a loss of performance due to the use of misnormalized data or design learning algorithms which are inherently capable of dealing with unnormalized data. In the field of optimization, it is a settled matter that algorithms should operate independent of an individual dimensions scaling [Oren, 1974]. The same structure defines natural gradients [Wagenaar, 1998] where in the stochastic setting, results indicate that for the parametric case the Fisher metric is the unique invariant metric satisfying a certain regular and monotone property [Corcuera and Giummole, 1998]. Our interest here is in the online learning setting, where this structure is rare: typically regret bounds depend on the norm of features.

The biggest practical benefit of invariance to feature scaling is that learning algorithms "just work" in a more general sense. This is of significant importance in online learning settings where fiddling with hyper-parameters is often common, and this work can be regarded as an alternative to investigations of optimal hyper-parameter tuning [Bergstra and Bengio, 2012, Snoek et al., 2012, Hutter et al., 2013]. With a normalized update users do not need to know (or remember) to pre-normalize input datasets and the need to worry about hyper-parameter tuning is greatly reduced. In practical experience, it is common for those unfamiliar with machine learning to create and attempt to use datasets without proper normalization.

Eliminating the need to normalize data also reduces computational requirements at both training and test time. For particularly large datasets this can become important, since the computational cost in time and RAM of doing normalization can rival the cost and time of doing the machine learning (or even worse for naive centering of sparse data). Similarly, for applications which are constrained by testing time, knocking out the need for feature normalization allows more computational performance with the same features or better prediction performance when using the freed computational resources to use more features.

1.1 Adversarial Scaling

Adversarial analysis is fairly standard in online learning. However, an adversary capable of rescaling features can induce unbounded regret in common gradient descent methods. As an example consider the standard regret bound for projected online convex subgradient descent after T rounds using the best learning rate in hindsight [Zinkevich, 2003],

$$R \leq \sqrt{T}||w^*||_2 \max_{t\in 1:T} ||g_t||_2.$$

Here w^* is the best predictor in hindsight and $\{g_t\}$ is the sequence of instantaneous gradients encountered by the algorithm. Suppose $w^* = (1, 1) \in \mathbb{R}^2$ and imagine scaling the first coordinate by a factor of s. As $s \to \infty$, $||w^*||_2$ ap-

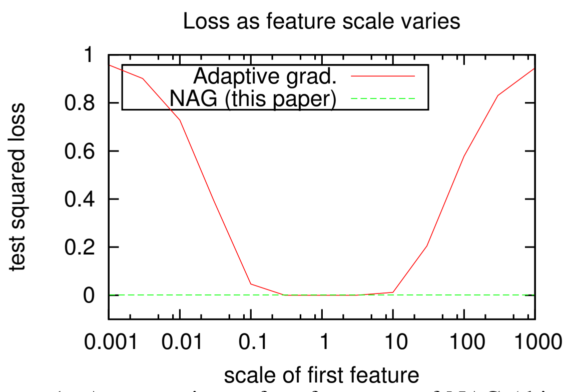


Figure 1: A comparison of performance of NAG (this paper) and adaptive gradient [McMahan and Streeter, 2010, Duchi et al., 2011] on a synthetic dataset with varying scale in the first feature.

proaches 1, but unfortunately for a linear predictor the gradient is proportional to the input, so $\max_{t\in 1:T} ||g_t||_2$ can be made arbitrarily large. Conversely as $s \to 0$, the gradient sequence remains bounded but $||w^*||_2$ becomes arbitrarily large. In both cases the regret bound can be made arbitrarily poor. This is a real effect rather than a mere artifact of analysis, as indicated by experiments with a synthetic two dimensional dataset in figure 1.1.

Adaptive first-order online methods [McMahan and Streeter, 2010, Duchi et al., 2011] also have this vulnerability, despite adapting the geometry to the input sequence. Consider a variant of the adaptive gradient update (without projection)

$$w_{t+1} = w_t - \eta \operatorname{diag}(\sum_{s=1}^{t} g_s g_s^T)^{-1/2} g_t,$$

which has associated regret bound of order

$$||w^*||_2\, d^{1/2} \sqrt{\inf_S \left\{ \sum_{t=1}^{T} \langle g_t, S^{-1} g_t \rangle : S \succeq 0, \operatorname{tr}(S) \le d \right\}}.$$

Again by manipulating the scaling of a single axis this can be made arbitrarily poor.

The online Newton step [Hazan, 2006] algorithm has a regret bound independent of units as we address here. Unfortunately ONS space and time complexity grows quadratically with the length of the input sequence, but the existence of ONS motivates the search for computationally viable scale invariant online learning rules.

Similarly, the second order perceptron [Cesa-Bianchi et al., 2005] and AROW [Crammer et al., 2009] partially address this problem for hinge loss. These algorithms are not unit-free because they have hyperparameters whose optimal value varies with the scaling of features and again have running times that are superlinear in the dimensionality. More recently, diagonalized second order perceptron and AROW have been proposed [Orabona et al., 2012]. These algorithms are linear time, but their analysis is generally not unit free since it explicitly depends on the norm of the weight vector. Corollary 3 is unit invariant. A comparative analysis of empirical performance would be interesting to observe.

The use of unit invariant updates have been implicitly studied with asymptotic analysis and empirics. For example [Schaul et al., 2012] uses a per-parameter learning rate proportional to an estimate of gradient squared divided by variance and second derivative. Relative to this work, we prove finite regret bound guarantees for our algorithm.

1.2 Contributions

We define normalized online learning algorithms which are invariant to feature scaling, then show that these are interesting algorithms theoretically and experimentally.

We define a scaling adversary for online learning analysis. The critical additional property of this adversary is that algorithms with bounded regret must have updates which are invariant to feature scale. We prove that our algorithm has a small regret against this more stringent adversary.

We then experiment with this learning algorithm on a number of datasets. For pre-normalized datasets, we find that it makes little difference as expected, while for unnormalized or improperly normalized datasets this update rule offers large advantages over standard online update rules. All of our code is a part of the open source Vowpal Wabbit project [Ross et al., 2012].

2 Notation

Throughout this draft, the indices i, j indicate elements of a vector, while the index t, T or a particular number indicates time. A label y is associated with some features x, and we are concerned with linear prediction $\sum_i w_i x_i$ resulting in some loss for which a gradient g can be computed with respect to the weights. Other notation is introduced as it is defined.

3 The algorithm

We start with the simplest version of a scale invariant online learning algorithm.

NG (Normalized Gradient Descent) is presented in algorithm 1. NG adds scale invariance to online gradient descent, making it work for any scaling of features within the dataset.

Without s, N, this algorithm simplifies to standard stochas-

Algorithm 1 NG(learning_rate η_t)

```
1. Initially $w_i = 0$, $s_i = 0$, $N = 0$
2. For each timestep $t$ observe example $(x, y)$
    (a) For each $i$, if $|x_i| > s_i$
        i. $w_i \leftarrow \frac{w_i s_i^2}{|x_i|^2}$
        ii. $s_i \leftarrow |x_i|$
    (b) $\hat{y} = \sum_i w_i x_i$
    (c) $N \leftarrow N + \sum_i \frac{x_i^2}{s_i^2}$
    (d) For each $i$,
        i. $w_i \leftarrow w_i - \eta_t \frac{t}{N} \frac{1}{s_i^2} \frac{\partial L(\hat{y}, y)}{\partial w_i}$
```

Algorithm 2 NAG(learning_rate η)

```
1. Initially $w_i = 0$, $s_i = 0$, $G_i = 0$, $N = 0$
2. For each timestep $t$ observe example $(x, y)$
    (a) For each $i$, if $|x_i| > s_i$
        i. $w_i \leftarrow \frac{w_i s_i}{|x_i|}$
        ii. $s_i \leftarrow |x_i|$
    (b) $\hat{y} = \sum_i w_i x_i$
    (c) $N \leftarrow N + \sum_i \frac{x_i^2}{s_i^2}$
    (d) For each $i$,
        i. $G_i \leftarrow G_i + \left(\frac{\partial L(\hat{y}, y)}{\partial w_i}\right)^2$
        ii. $w_i \leftarrow w_i - \eta \sqrt{\frac{t}{N}} \frac{1}{s_i \sqrt{G_i}} \frac{\partial L(\hat{y}, y)}{\partial w_i}$
```

tic gradient descent.

The vector element s_i stores the magnitude of feature i according to $s_{ti} = \max_{t' \in \{1 \ldots t\}} |x_{t'i}|$. These are updated and maintained online in steps 2.(a).ii, and used to rescale the update on a per-feature basis in step 2.(d).i.

Using N makes the learning rate (rather than feature scale) control the average change in prediction from an update. Here N/t is the average change in the prediction excluding η, so multiplying by $1/(N/t) = t/N$ causes the average change in the prediction to be entirely controlled by η.

Step 2.(a).i squashes a weight i when a new scale is encountered. Neglecting the impact of N, the new value is precisely equal to what the weight's value would have been if all previous updates used the new scale.

Many other online learning algorithms can be made scale invariant using variants of this approach. One attractive choice is adaptive gradient descent [McMahan and Streeter, 2010, Duchi et al., 2011] since this also has per-feature learning rates. The normalized version of adaptive gradient descent is given in algorithm 2.

In order to use this, the algorithm must maintain the sum of gradients squared $G_i = \sum_{(x,y) \text{ observed}} \left(\frac{\partial L(\hat{y}, y)}{\partial w_i}\right)^2$ for feature i in step 2.d.i. The interaction between N and G is somewhat tricky, because a large average update (i.e. most features have a magnitude near their scale) increases the value of G_i as well as N implying the power on N must be decreased to compensate. Similarly, we reduce the power on s_i and $|x_i|$ to 1 throughout. The more complex update rule is scale invariant and the dependence on N introduces an automatic global rescaling of the update rule.

In the next sections we analyze and justify this algorithm. We demonstrate that NAG competes well against a set of predictors w with predictions ($w^\top x$) bounded by some constant over all the inputs x_t seen during training. In practice, as this is potentially sensitive to outliers, we also consider a squared norm version of NAG, which we refer to as sNAG that is a straightforward modification—we simply keep the accumulator $s_i = \sum x_i^2$ and use $\sqrt{s_i/t}$ in the update rule. That is, normalization is carried using the standard deviation (more precisely, the square root of the second moment) of each feature, rather than the max norm. With respect to our analysis below, this simple modification can be interpreted as changing slightly the set of predictors we compete against, i.e. predictors with predictions bounded by a constant only over the inputs within 1 standard deviation. Intuitively, this is more robust and appropriate in the presence of outliers. While our analysis focuses on NAG, in practice, sNAG sometimes yield improved performance.

4 The Scaling Adversary Setting

In common machine learning practice, the choice of units for any particular feature is arbitrary. For example, when estimating the value of a house, the land associated with a house may be encoded either in acres or square feet. To model this effect, we propose a scaling adversary, which is more powerful than the standard adversary in adversarial online learning settings.

The setting for analysis is similar to adversarial online linear learning, with the primary difference in the goal. The setting proceeds in the following round-by-round fashion where

1. Prior to all rounds, the adversary commits to a fixed positive-definite matrix S. This is not revealed to the learner.
2. On each round t,
    (a) The adversary chooses a vector x_t such that $||S^{1/2} x_t||_\infty \leq 1$, where $S^{1/2}$ is the principal

square root.

(b) The learner makes a prediction $\hat{y}_t = w_t^\top x_t$.

(c) The correct label y_t is revealed and a loss $\ell(\hat{y}_t, y_t)$ is incurred.

(d) The learner updates the predictor to w_{t+1}.

For example, in a regression setting, ℓ could be the squared loss $\ell(\hat{y}, y) = (\hat{y} - y)^2$, or in a binary classification setting, ℓ could be the hinge loss $\ell(\hat{y}, y) = \max(0, 1 - y\hat{y})$. We consider general cases where the loss is only a function of $\hat{y}$ (i.e. no direct penalty on w) and convex in $\hat{y}$ (therefore convex in w).

Although step 1 above is phrased in terms of an adversary, in practice what is being modeled is "the data set was prepared using arbitrary units for each feature."

Step 2 (a) above is phrased in terms of ∞-norm for ease of exposition, but more generally can be considered any p-norm. Additionally, this step can be generalized to impose a different constraint on the inputs. For instance, instead requiring all points lie inside some p-norm ball, we could require that the second moment of the inputs, under some scaling matrix S is 1. This is the model of the adversary for sNAG.

4.1 Competing against a Bounded Output Predictor

Our goal is to compete against the set of weight vectors whose output is bounded by some constant C over the set of inputs the adversary can choose. Given step 2 (a) above, this is equivalent to $\mathcal{W}_X^C = \{w \mid ||S^{-1/2}w||_1 \le C\}$, i.e., the set of w with dual norm less than C. In other words, the regret R_T at timestep T is given by:

$$R_T = \sum_{t=1}^{T} \ell(\hat{y}_t, y_t) - \min_{w|\forall t, w^\top x_t \le C} \sum_{t=1}^{T} \ell(w^\top x_t, y_t)$$

Here we use the fact that $\{w^\top x_t \le C\} = \{w \mid ||S^{-1/2}w||_1 \le C\}$. In the more general case of a p-norm for step 2 (a), we would choose $\mathcal{W}_X^C = \{w \mid ||S^{-1/2}w||_q \le C\}$ for q such that $\frac{1}{p} + \frac{1}{q} = 1$. Note that the "true" S of the adversary is an abstraction. It is unknown and only partially revealed through the data. In our analysis, we will instead be interested to bound regret against bounded output predictors for an empirical estimate of S, defined by the minimum volume L_p ball containing all observed inputs. For $p = \infty$, $\mathcal{W}_X^C$ for the "true" S is always a subset of the predictors allowed under this empirical S (assuming both are diagonal matrices). In general, this does not necessarily hold for all p norms, but the empirical S always allows a larger volume of predictors than the "true" S.

5 Analysis

In this section, we analyze scale invariant update rules in several ways. The analysis is structurally similar to that used for adaptive gradient descent [McMahan and Streeter, 2010, Duchi et al., 2011] with necessary differences to achieve scale invariance. We analyze the best solution in hindsight, the best solution in a transductive setting, and the best solution in an online setting. These settings are each a bit more difficult than the previous, and in each we prove regret bounds which are invariant to feature scales.

We consider algorithms updating according to $w_{t+1} = w_t - A_t^{-1} g_t$, where g_t is the gradient of the loss at time t w.r.t. w at w_t, and A_t is a sequence of $d \times d$ symmetric positive (semi-)definite matrices that our algorithm can choose. Both algorithms 1 and 2 fit this general framework. Combining the convexity of the loss function and the definition of the update rule yields the following result.

Lemma 1.

$$\begin{aligned} 2R_T \le {} & (w_1 - w^*)A_1(w_1 - w^*) \\ & + \sum_{t=1}^{T} (w_t - w^*)^\top (A_{t+1} - A_t)(w_t - w^*) \\ & + \sum_{t=1}^{T} g_t^\top A_t^{-1} g_t. \end{aligned}$$

We defer all proofs to the appendix.

5.1 Best Choice of Conditioner in Hindsight

Suppose we start from $w_1 = 0$ and before the start of the algorithm, we would try to guess what is the best fixed matrix A, so that $A_t = A$ for all t. In order to minimize regret, what would the best guess be? This was initially analyzed for adaptive gradient descent [McMahan and Streeter, 2010, Duchi et al., 2011]. Consider the case where A is a diagonal matrix.

Using lemma 1, for a fixed diagonal matrix A and with $w_1 = 0$, the regret bound is:

$$R_T \le \frac{1}{2} \sum_{i=1}^{d} \left(A_{ii}(w_i^*)^2 + \sum_{t=1}^{T} \frac{g_{ti}^2}{A_{ii}} \right).$$

Taking the derivative w.r.t. A_{ii}, we obtain:

$$\begin{aligned} & \frac{\partial}{\partial A_{ii}} \frac{1}{2} \sum_{i=1}^{d} \left(A_{ii}(w_i^*)^2 + \sum_{t=1}^{T} \frac{g_{ti}^2}{A_{ii}} \right) \\ & = \frac{1}{2} \left((w_i^*)^2 - \sum_{t=1}^{T} \frac{g_{ti}^2}{A_{ii}^2} \right). \end{aligned}$$

Solving for when this is 0, we obtain

$$A_{ii}^* = \frac{\sqrt{\sum_{t=1}^{T} g_{ti}^2}}{|w_i^*|}.$$

For this particular choice of A, then the regret is bounded by

$$R_T \leq \sum_{i=1}^{d} \left(|w_i^*| \sqrt{\sum_{t=1}^{T} g_{ti}^2} \right).$$

We can observe that this regret is the same no matter the scaling of the inputs. For instance if any axis i is scaled by a factor s_i, then w_i^* would be a factor $1/s_i$ smaller, and the gradient g_{ti} a factor s_i larger, which would cancel out. Hence this regret can be thought as the regret the algorithm would obtain when all features have the same unit scale.

However, because of the dependency of A on w^*, this does not give us a good way to approximate this with data we have observed so far. To remove this dependency, we can analyze for the best A when assuming a worst case for w^*. This is the point at which the analysis here differs from adaptive gradient descent where the dependence on w^* was dropped.

Lemma 2. *Let S be the diagonal matrix with minimum determinant (volume) s.t. $||S^{1/2}x_i||_p \leq 1$ for all $i \in 1:T$. The solution to*

$$\min_{A} \max_{w^* \in \mathcal{W}_X^C} \frac{1}{2} \sum_{i=1}^{d} \left(A_{ii}(w_i^*)^2 + \sum_{t=1}^{T} \frac{g_{ti}^2}{A_{ii}} \right)$$

is given by

$$A_{ii}^* = \frac{1}{C} \sqrt{\frac{\sum_{t=1}^{T} g_{ti}^2}{S_{ii}}},$$

and the regret bound for this particular choice of A is given by

$$R_T \leq C \sum_{i=1}^{d} \sqrt{S_{ii} \sum_{t=1}^{T} g_{ti}^2}.$$

Again the value of the regret bound does not change if the features are rescaled. This is most easily appreciated by considering a specific norm. The simplest case is for $p = \infty$ where the coefficients S_{ii} can be defined directly in terms of the range of each feature, i.e. $S_{ii} = \frac{1}{\max_t |x_{ti}|^2}$. Thus for $p = \infty$, we can choose

$$A_{ii}^* = \frac{1}{C} \sqrt{\sum_{t=1}^{T} g_{ti}^2 \max_{t \in 1:T} |x_{ti}|},$$

leading to a regret of

$$R_T \leq C \sum_{i=1}^{d} \frac{\sqrt{\sum_{t=1}^{T} g_{ti}^2}}{\max_{t \in 1:T} |x_{ti}|}.$$

The scale invariance of the regret bound is now readily apparent. This regret can potentially be order $O(d\sqrt{T})$.

5.2 $p = 2$ case

For $p = 2$, computing the coefficients S_{ii} is more complicated, but if you have access to the actual coefficients S_{ii}, the regret is order $O(\sqrt{dT})$. This can be seen as follows. Let $g'_t = \frac{\partial \ell}{\partial \hat{y}}\Big|_{\hat{y}_t, y_t}$ the derivative of the loss at time t evaluated at the predicted $\hat{y}_t$. Then $g_{ti} = g'_t x_{ti}$ and we can see that:

$$\begin{aligned}
&\textstyle\sum_{i=1}^{d} \sqrt{S_{ii} \sum_{t=1}^{T} g_{ti}^2} \\
&= d \textstyle\sum_{i=1}^{d} \frac{1}{d} \sqrt{S_{ii} \sum_{t=1}^{T} g_{ti}^2} \\
&\leq d \sqrt{\textstyle\sum_{i=1}^{d} \frac{1}{d} S_{ii} \sum_{t=1}^{T} g_{ti}^2} \\
&= \sqrt{d} \sqrt{\textstyle\sum_{i=1}^{d} S_{ii} \sum_{t=1}^{T} g_{ti}^2} \\
&= \sqrt{d} \sqrt{\textstyle\sum_{t=1}^{T} g_t'^2 \sum_{i=1}^{d} S_{ii} x_{ti}^2} \\
&\leq \sqrt{d} \sqrt{\textstyle\sum_{t=1}^{T} g_t'^2}
\end{aligned}$$

where the last inequality holds by assumption. For $p = 2$, we have $R_T \leq C\sqrt{d}\sqrt{\sum_{t=1}^{T} g_t'^2}$.

5.3 Adaptive Conditioner

Lemma 2 does not lead to a practical algorithm, since at time t, we only observed $g_{1:t}$ and $x_{1:t}$, when performing the update for w_{t+1}. Hence we would not be able to compute this optimal conditioner A^*. However it suggests that we could potentially approximate this ideal choice using the information observed so far, e.g.,

$$A_{t,ii} = \frac{1}{C} \sqrt{\frac{\sum_{s=1}^{t} g_{si}^2}{S_{ii}^{(t)}}}, \tag{1}$$

where $S^{(t)}$ is the diagonal matrix with minimum determinant s.t. $||S^{1/2}x_i||_p \leq 1$ for all $i \in 1:t$. There are two potential sources of additional regret in the above choice, one from truncating the sum of gradients, and the other from estimating the enclosing volume online.

5.3.1 Transductive Case

To demonstrate that truncating the sum of gradients has only a modest impact on regret we first consider the transductive case, i.e., we assume we have access to all inputs $x_{1:T}$ that are coming in advance. However at time t, we do not know the future gradients $g_{t+1:T}$. Hence for this setting we could consider a 2-pass algorithm. On the first pass, compute the diagonal matrix S, and then on the second pass, perform adaptive gradient descent with the following conditioner at time t:

$$A_{t,ii} = \frac{1}{C\eta} \sqrt{\frac{\sum_{j=1}^{t} g_{ji}^2}{S_{ii}}}. \tag{2}$$

We would like to be able to show that if we adapt the conditioner in this way, than our regret is not much worse than with the best conditioner in hindsight. To do so, we must introduce a projection step into the algorithm. The projection step enables us to bound the terms in lemma 1 corresponding to the use of a non-constant conditioner, which are related to the maximum distance between an intermediate weight vector and the optimal weight vector.

Define the projection $\Pi^A_{S,C,q}$ as

$$\Pi^A_{S,C,q}(w') = \operatorname*{arg\,min}_{w\in\mathbb{R}^d \,|\, ||S^{-1/2}w||_q \le C} (w-w')^\top A(w-w').$$

Utilizing this projection step in the update we can show the following.

Theorem 1. *Let S be the diagonal matrix with minimum determinant s.t. $||S^{1/2}x_i||_p \le 1$ for all $i \in 1:T$, and let $1/q = 1 - 1/p$. If we choose A_t as in Equation 2 with $\eta = \sqrt{2}$ and use projection $w_{t+1} = \Pi^{A_t}_{S,C,q}(w_t - A_t^{-1}g_t)$ at each step, the regret is bounded by*

$$R_T \le 2C\sqrt{2}\sum_{i=1}^d \sqrt{S_{ii}\sum_{j=1}^T g_{ji}^2}.$$

We note that this is only a factor $2\sqrt{2}$ worse than when using the best fixed A in hindsight, knowing all gradients in advance.

5.3.2 Streaming Case

In this section we focus on the case $p = \infty$.

The transductive analysis indicates that using a partial sum of gradients does not meaningfully degrade the regret bound. We now investigate the impact of estimating the enclosing ellipsoid with a diagonal matrix online using only observed inputs,

$$A_{t,ii} = \frac{1}{C\eta}\sqrt{\sum_{j=1}^t g_{ji}^2}\max_{j\in 1:t}|x_{ji}|. \qquad (3)$$

The diagonal approximation is necessary for computational efficiency in NAG.

Intuitively the worst case is when the conditioner in equation 3 differs substantially from the transductive conditioner of equation 2 over most of the sequence. This is reflected in the regret bound below which is driven by the ratio between the first non-zero value of an input x_{ji} encountered in the sequence and the maximum value it obtains over the sequence.

Theorem 2. *Let $p = \infty$, $q = 1$, and let $S^{(t)}$ be the diagonal matrix with minimum determinant s.t. $||S^{1/2}x_i||_p \le 1$ for all $i \in 1:t$. Let $\Delta_i = \frac{\max_{t\in 1:T}|x_{ti}|}{|x_{t_0^i i}|}$, for t_0^i the first timestep the i^{th} feature is non-zero. If we choose A_t as in Equation 3, $\eta = \sqrt{2}$ and use projection $w_{t+1} = \Pi^{A_t}_{S^{(t)},C,q}(w_t - A_t^{-1}g_t)$ at each step, the regret is bounded by*

$$R_T \le C\sum_{i=1}^d \frac{\sqrt{\sum_{j=1}^T g_{ji}^2}}{\max_{j\in 1:T}|x_{ji}|}\left(\frac{1+6\Delta_i+\Delta_i^2}{2\sqrt{2}}\right).$$

Comparing theorem 2 with theorem 1 reveals the degradation in regret due to online estimation of the enclosing ellipsoid. Although an adversary can in general manipulate this to cause large regret, there are nontrivial cases for which theorem 2 provides interesting protection. For example, if the non-zero feature values for dimension i range over $[s_i, 2s_i]$ for some unknown s_i, then $1 \le \Delta_i \le 2$ and the regret bound is only a constant factor worse than the best choice of conditioner in hindsight.

Because the worst case streaming scenario is when the initial sequence has much lower scale than the entire sequence, we can improve the bound if we weaken the ability of the adversary to choose the sequence order. In particular, we allow the adversary to choose the sequence $\{x_t, y_t\}_{t=1}^T$ but then we subject the sequence to a random permutation before processing it. We can show that with high probability we must observe a high percentile value of each feature after only a few datapoints, which leads to the following corollary to theorem 2.

Corollary 1. *Let $\{x_t, y_t\}_{t=1}^T$ be an exchangeable sequence with $x_t \in \mathbb{R}^d$. Let $p = \infty$, $q = 1$, and let $S^{(t)}$ be the diagonal matrix with minimum determinant s.t. $||S^{1/2}x_i||_p \le 1$ for all $i \in 1:t$. Choose $\delta > 0$ and $\nu \in (0,1)$. Let $\Delta_i = \frac{\max_{t\in 1:T}|x_{ti}|}{\max_{t\in 1:\tau}|x_{ti}|}$, where*

$$\tau = \left\lceil \frac{\log(d/\delta)}{\nu} \right\rceil.$$

If $R_{\max}$ is the maximum regret that can be incurred on a single example, then choosing $\eta = \sqrt{2}$, and using projection $w_{t+1} = \Pi^{A_t}_{S^{(t)},C,q}(w_t - A_t^{-1}g_t)$ at each step, the regret is bounded by

$$\begin{aligned} R_T \le\ & \left\lfloor \tfrac{\log(d/\delta)}{\nu} \right\rfloor R_{\max} \\ & + C\textstyle\sum_{i=1}^d \frac{\sqrt{\sum_{j=1}^T g_{ji}^2}}{\max_{j\in 1:T}|x_{ji}|}\left(\frac{1+6\Delta_i+\Delta_i^2}{2\sqrt{2}}\right), \end{aligned}$$

and with probability at least $(1-\delta)$ over sequence realizations, for all $i \in 1:d$,

$$\Delta_i \le \frac{\max_{t\in 1:T}|x_{ti}|}{\text{Quantile}\left(\{|x_{ti}|\}_{t=1}^T, 1-\nu\right)},$$

where Quantile$(\cdot, 1-\nu)$ *is the $(1-\nu)$-th quantile of a given sequence.*

The quantity $R_{\max}$ can be related to C, if when making predictions, we always truncate $w_t^\top x_t$ in the interval $[-C, C]$. For instance, for the hinge loss and logistic loss,

Dataset	Size	Features	Scale Range
Bank	45,212	7	[31, 102127]
Census	199,523	13	[2, 99999]
Covertype	581,012	54	[1, 7173]
CT Slice	53,500	360	[0.158, 1]
MSD	463,715	90	[60, 65735]
Shuttle	43,500	9	[105, 13839]

Table 1: Datasets used for experiments. CT Slice and MSD are regression tasks, all others are classification. The scale of a feature is defined as the maximum empirical absolute value, and the scale range of a dataset defined as the minimum and maximum feature scales.

$R_{\max} \leq C+1$ if we truncate our predictions this way. Similarly for the squared loss, $R_{\max} \leq 4C \max(C, \max_t |y_t|)$. Although in theory an adversary can manipulate the ratio between the maximum and an extreme quantile to induce arbitrarily bad regret (i.e. make $\frac{\max_{t \in 1:T} |x_{ti}|}{\text{Quantile}(\{|x_{ti}|\}_{t=1}^T, 1-\nu)}$ arbitrarily large even for small ν), in practice we can often expect this quantity to be close to 1[1], and thus corollary 1 suggests that we may perform not much worse than when the scale of the features are known in advance. Our experiments demonstrate that this is the common behavior of the algorithm in practice.

6 Experiments

Table 2 compares a variant of the normalized learning rule to the adaptive gradient method [McMahan and Streeter, 2010, Duchi et al., 2011] with $p = 2$ and without projection step for both algorithms. For each data set we exhaustively searched the space of learning rates to optimize average progressive validation loss. Besides the learning rate, the learning rule was the only parameter adjusted between the two conditions. The loss function used depended upon the task associated with the dataset, which was either 0-1 loss for classification tasks or squared loss for regression tasks. For regression tasks, the loss is divided by the worst possible squared loss, i.e., $(\max - \min)^2$.

The datasets utilized are: *Bank*, the UCI [Frank and Asuncion, 2010] Bank Marketing Data Set [Moro et al., 2011]; *Census*, the UCI Census-Income KDD Data Set; *Covertype*, the UCI Covertype Data Set; *CT Slice*, the UCI Relative Location of CT Slices on Axial Axis Data Set; *MSD*, the Million Song Database [Bertin-Mahieux et al., 2011]; and *Shuttle*, the UCI Statlog Shuttle Data Set. These were selected as

[1]For instance, if $|x_{ti}|$ are exponentially distributed, Δ_i is roughly less than $\log(T/\delta)/\log(1/\nu)$ with probability at least $1-\delta$, thus choosing $\nu = T^{-\alpha}$, for $\alpha \in (0, 0.5]$ makes this a small constant of order $\alpha^{-1}\log(1/\delta)$, while keeping the first term involving $R_{\max}$ order of $T^{\alpha} \leq \sqrt{T}$.

Dataset	NAG		AG	
	η^*	Loss	η^*	Loss
Bank	0.55	**0.098**	$5.5 * 10^{-5}$	0.109
(Maxnorm)	0.55	0.099	0.061	0.099
Census	0.2	**0.050**	$1.2 * 10^{-6}$	0.054
(Maxnorm)	0.25	0.050	$8.3 * 10^{-3}$	0.051
Covertype	1.5	**0.27**	$5.6 * 10^{-7}$	0.32
(Maxnorm)	1.5	0.27	0.2	0.27
CT Slice	2.7	0.0023	0.022	0.0023
(Maxnorm)	2.7	0.0023	0.022	0.0023
MSD	9.0	**0.0110**	$5.5 * 10^{-7}$	0.0130
(Maxnorm)	9.0	0.0110	6.0	0.0108
Shuttle	7.4	0.036	$7.5 * 10^{-4}$	0.040
(Maxnorm)	7.4	0.036	16.4	0.035

Table 2: Comparison of NAG with Adaptive Gradient (AG) across several data sets. For each data set, the first line in the table contains the results using the original data, and the second line contains the results using a max-norm pre-normalized version of the original data. For both algorithms, η^* is the optimal in-hindsight learning rate for minimizing progressive validation loss (empirically determined). Significance (bolding) was determined using a relative entropy Chernoff bound with a 0.1 probability of bound failure.

Dataset	sNAG		AG	
	η^*	Loss	η^*	Loss
Bank	0.3	**0.098**	$5.5 * 10^{-5}$	0.109
(Sq norm)	0.3	0.098	0.033	0.097
Census	0.11	**0.050**	$1.2 * 10^{-6}$	0.054
(Sq norm)	0.11	0.050	$1.6 * 10^{-3}$	0.048
Covertype	2.2	**0.28**	$5.6 * 10^{-7}$	0.32
(Sq norm)	2.7	0.28	0.04	0.28
CT Slice	2.7	**0.0019**	0.022	0.0023
(Sq norm)	2.7	0.0019	0.0067	0.0019
MSD	7.4	0.0119	$5.5 * 10^{-7}$	0.0130
(Sq norm)	7.4	0.0118	0.05	0.0120
Shuttle	11	**0.026**	$7.5 * 10^{-4}$	0.040
(Sq norm)	9	0.026	0.818	0.026

Table 3: (Online Mean Square Normalized) Comparison of NAG with Adaptive Gradient (AG) across several data sets. For each data set, the first line in the table contains the results using the original data, and the second line contains the results using a squared-norm pre-normalized version of the original data. For both algorithms, η^* is the optimal in-hindsight learning rate for minimizing progressive validation loss (empirically determined). Significance (bolding) was determined using a relative entropy Chernoff bound with a 0.1 probability of bound failure.

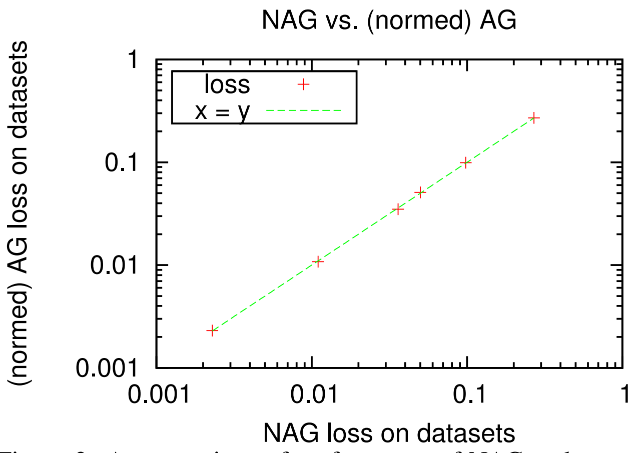


Figure 2: A comparison of performance of NAG and pre-normed AG. The results are identical, indicating that NAG effectively obsoletes pre-normalization.

public datasets plausibly exhibiting varying scales or lack of normalization. On other pre-normalized datasets which are publicly available, we observed relatively little difference between these update rules. To demonstrate the effect of pre-normalization on these data sets, we constructed a pre-normalized version of each one by dividing every feature by its maximum empirical absolute value.

Some trends are evident from table 2. First, the normalized learning rule (as expected) has highest impact when the individual feature scales are highly disparate, such as data assembled from heterogeneous sensors or measurements. For instance, the CT slice data set exhibits essentially no difference; although CT slice contains physical measurements, they are histograms of raw readings from a single device, so the differences between feature ranges is modest (see table 1). Conversely the Covertype dataset shows a 5% decrease in multiclass 0-1 loss over the course of training. Covertype contains some measurements in units of meters and others in degrees, several "hillshade index" values that range from 0 to 255, and categorical variables.

The second trend evident from table 2 and reproduced in figure 3 is that the optimal learning rate is both closer to 1 in absolute terms, and varies less in relative terms, between data sets. This substantially eases the burden of tuning the learning rate for high performance. For example, a randomized search [Bergstra and Bengio, 2012] is much easier to conduct given that the optimal value is extremely likely to be within $[0.01, 10]$ independent of the data set.

The last trend evident from table 2 is the typical indifference of the normalized learning rate to pre-normalization, specifically the optimal learning rate and resulting progressive validation loss. In addition pre-normalization effectively eliminates the difference between the normalized and Adagrad updates, indicating that the online algorithm

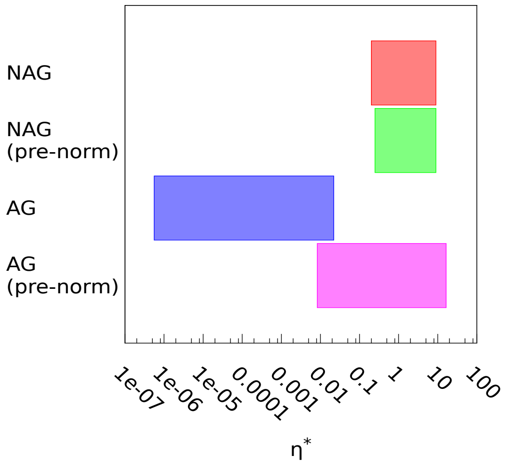


Figure 3: Each color represents the range of optimal in-hindsight learning rates η^* across the datasets for the different learning algorithms. NAG exhibits a much smaller range of optimal values even when the data sets are pre-normalized, easing the problem of hyperparameter selection.

achieves results similar to the transductive algorithm for max norm.

For comprehensiveness, we also compared sNAG with a squared norm pre-normalizer, and found the story much the same in table 3. In particular sNAG dominated AG on most of the datasets and performed similarly to AG when the data was pre-normalized with a squared norm (standard deviation). It is also interesting to observe that sNAG performs slightly better than NAG on a few datasets, agreeing with our intuition that it should be more robust to outliers. Empirically, sNAG appears somewhat more robust than NAG at the cost of somewhat more computation.

7 Summary

We evaluated performance of Normalized Adaptive Gradient (NAG) on the most difficult unnormalized public datasets available and found that it provided performance similar to Adaptive Gradient (AG) applied to pre-normalized datasets while simultaneously collapsing the range hyperparameter search required to achieve good performance. Empirically, this makes NAG a capable and reliable learning algorithm.

We also defined a scaling adversary and proved that our algorithm is robust and efficient against this scaling adversary unlike other online learning algorithms.

Acknowledgements

We would like to thank Miroslav Dudik for helpful discussions.

References

[Allwein et al., 2001] Allwein, E., Schapire, R., and Singer, Y. (2001). Reducing multiclass to binary: A unifying approach for margin classifiers. *The Journal of Machine Learning Research*, 1:113–141.

[Bergstra and Bengio, 2012] Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305.

[Bertin-Mahieux et al., 2011] Bertin-Mahieux, T., Ellis, D. P., Whitman, B., and Lamere, P. (2011). The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*.

[Cesa-Bianchi et al., 2005] Cesa-Bianchi, N., Conconi, A., and Gentile, C. (2005). A second-order perceptron algorithm. *SIAM Journal on Computing*, 34(3):640–668.

[Corcuera and Giummole, 1998] Corcuera, J. M. and Giummole, F. (1998). A characterization of monotone and regular divergences. *Annals of the Institute of Statistical Mathematics*, 50(3):433–450.

[Crammer et al., 2009] Crammer, K., Kulesza, A., and Dredze, M. (2009). Adaptive regularization of weight vectors. In *Advances in Neural Information Processing Systems*.

[Duchi et al., 2011] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methdos for online learning and stochastic optimization. *JMLR*.

[Frank and Asuncion, 2010] Frank, A. and Asuncion, A. (2010). UCI machine learning repository.

[Hazan, 2006] Hazan, E. (2006). Efficient algorithms for online convex optimization and their applications. Technical report, Princeton.

[Hutter et al., 2013] Hutter, F., Hoos, H., and Leyton-Brown, K. (2013). Identifying key algorithm parameters and instance features using forward selection. In *Learning and Intelligent Optimization (LION 7)*.

[Li and Zhang, 1998] Li, G. and Zhang, J. (1998). Sphering and its properties. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 119–133.

[Loosli et al., 2007] Loosli, G., Canu, S., and Bottou, L. (2007). Training invariant support vector machines using selective sampling. In Bottou, L., Chapelle, O., DeCoste, D., and Weston, J., editors, *Large Scale Kernel Machines*, pages 301–320. MIT Press, Cambridge, MA.

[McMahan and Streeter, 2010] McMahan, H. B. and Streeter, M. (2010). Adaptive bound optimization for online convex optimization. In *Conference on Learning Theory*.

[Moro et al., 2011] Moro, S., Laureano, R., and Cortez, P. (2011). Using data mining for bank direct marketing: An application of the crisp-dm methodology. In et al., P. N., editor, *Proceedings of the European Simulation and Modelling Conference - ESM'2011*, pages 117–121, Guimaraes, Portugal. EUROSIS.

[Orabona et al., 2012] Orabona, F., Crammer, K., and Cesa-Bianchi, N. (2012). A generalized online mirror descent with applications to classification and regression. Technical report, Unimi.

[Oren, 1974] Oren, S. S. (1974). Self-scaling variable metric (ssvm) algorithms. part ii: Implementation and experiments. *Management Science*, 20(5):pp. 863–874.

[Ross et al., 2012] Ross, S., Mineiro, P., and Langford, J. (2012). Vowpal wabbit implementation of nag and snag. Technical report, github.com.

[Schaul et al., 2012] Schaul, T., Zhang, S., and LeCun, Y. (2012). No more pesky learning rates. *CoRR*, abs/1206.1106.

[Snoek et al., 2012] Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In *NIPS*.

[Sonnenburg and Franc, 2010] Sonnenburg, S. and Franc, V. (2010). COFFIN: a computational framework for linear SVMs. In *Proceedings of the 27nd International Machine Learning Conference*.

[Wagenaar, 1998] Wagenaar, D. (1998). Information geometry for neural networks. Technical report, King's College London.

[Zinkevich, 2003] Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the International Conference on Machine Learning (ICML 2003)*, pages 928–936.

Beyond Log-Supermodularity: Lower Bounds and the Bethe Partition Function

Nicholas Ruozzi
Communication Theory Laboratory
École Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
nicholas.ruozzi@epfl.ch

Abstract

A recent result has demonstrated that the Bethe partition function always lower bounds the true partition function of binary, log-supermodular graphical models. We demonstrate that these results can be extended to other interesting classes of graphical models that are not necessarily binary or log-supermodular: the ferromagnetic Potts model with a uniform external field and its generalizations and special classes of weighted graph homomorphism problems.

1 Introduction

A standard inference problem is to compute the partition function, or normalizing constant, of a given graphical model. As computing the partition function of an arbitrary graphical model is NP-hard, the partition function is often replaced by a more tractable approximation. A popular approximation to the partition function, due to its relationship to the belief propagation algorithm and its practical performance, is given by the Bethe partition function from statistical physics. However, the relationship between the Bethe partition function and the true partition function is difficult to characterize for an arbitrary graphical model.

Using a combinatorial characterization of the Bethe partition function from (Vontobel, 2013), we recently demonstrated that there exist nice families of graphical models for which the Bethe partition function provably lower bounds the true partition function (Ruozzi, 2012). Specifically, for binary graphical models, whenever the potential functions of the graphical model are all log-supermodular, the Bethe partition function always lower bounds the true partition function. As an example, the partition function of the ferromagnetic Ising model with an arbitrary external field can be expressed as the partition function of a log-supermodular function. In a very technical sense, these results can be extended beyond binary graphical models to other models over finite distributive lattices (e.g., any finite totally ordered set) as every finite distributive lattice is isomorphic to a sublattice of the Boolean lattice over $\{0, 1\}^n$ for some n (Alon and Spencer, 2000). However, natural candidates for non-binary graphical models, such as the ferromagnetic Potts model, for which one might suspect that the Bethe partition function again provides a lower bound are not log-supermodular in this sense.

In this work, we show that the results of Ruozzi (2012) can be extended to provide bounds on the Bethe partition function of other, not necessarily binary or log-supermodular, graphical models. Specifically, we show that the partition functions of certain graphical models can be equivalently expressed as the partition functions of log-supermodular graphical models over possibly different factor graphs and state spaces. Under certain conditions, this log-supermodular transformation can then be exploited to prove that, again, the Bethe partition function always provides a lower bound on the true partition function.

The models considered in this work include the ferromagnetic Potts model (with some restrictions on the choice of external field), generalizations of the ferromagnetic Potts model to matroids, certain weight enumerators of linear codes, and a subset of graphical models for the weighted graph homomorphism problem. For these models, we demonstrate that the Bethe partition function always provides a lower bound on the true partition function that is provably tighter than the lower bound corresponding to the naïve mean-field partition function.

This paper is organized as follows. In Section 2, we review the relevant background material: graphical models, graph covers, and approximate partition functions. In Section 4, we motivate the results in this work by looking at the simple case of pairwise binary graphical models. In Section 5, we show that the Bethe free energy provides a lower bound on the partition function of the ferromagnetic Potts model with a uniform external field, demonstrating by counter example that the results do not hold for arbitrary external fields. In addition, we show that similar results are true for several common generalizations of the ferromagnetic Potts model that include certain weight enumerators of linear codes. In Section 6, we consider the problem of counting weighted graph homomorphisms and demonstrate that the Bethe partition function provides a lower bound under certain restrictions on the target graph. Finally, we conclude with a short discussion in Section 7.

2 Graphical Models

Let $f : \mathcal{X}^n \to \mathbb{R}_{\geq 0}$ be a non-negative function where $\mathcal{X}$ is a finite set. A function f factors with respect to a hypergraph $G = (V, \mathcal{A})$, if there exist potential functions $\phi_i : \mathcal{X} \to \mathbb{R}_{\geq 0}$ for each $i \in V$ and $\psi_\alpha : \mathcal{X}^{|\alpha|} \to \mathbb{R}_{\geq 0}$ for each $\alpha \in \mathcal{A}$ such that

$$f(x_1, \ldots, x_n) = \prod_{i \in V} \phi_i(x_i) \prod_{\alpha \in \mathcal{A}} \psi_\alpha(x_\alpha).$$

The graph G together with the collection of potential functions ϕ and ψ define a graphical model that we will denote as $(G; \phi, \psi)$. For a given graphical model $(G; \phi, \psi)$, we are interested in computing the partition function

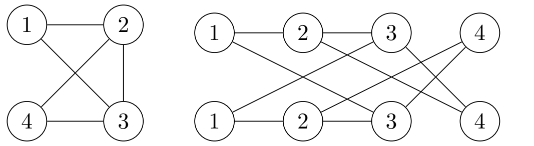
$$Z(G; \phi, \psi) = \sum_{x \in \mathcal{X}^{|V|}} \Big[\prod_{i \in V} \phi_i(x_i) \prod_{\alpha \in \mathcal{A}} \psi_\alpha(x_\alpha) \Big].$$

In general, computing the partition function is an NP-hard problem, but in practice, local message-passing algorithms based on approximations from statistical physics, such as loopy belief propagation, produce reasonable estimates in many settings.

2.1 Graph Covers

Graph covers have played an important role in our understanding of inference in graphical models (Vontobel, 2013; Vontobel and Koetter, 2005), and in particular, they are intimately related to the approximations of the partition function that we will consider in this work. Roughly speaking, if a graph H covers a graph G, then H looks locally the same as G.

Definition 2.1. *A graph H **covers** a graph $G = (V, E)$ if there exists a graph homomorphism $h : H \to G$ such that for all vertices $i \in G$ and all $j \in h^{-1}(i)$, h maps the neighborhood ∂j of j in H bijectively to the neighborhood ∂i of i in G.*



(a) A graph, G. (b) One possible cover of G.

Figure 1: An example of a graph cover. The nodes in the cover are labeled for the node that they copy in the base graph.

If $h(j) = i$, then we say that $j \in H$ is a copy of $i \in G$. Further, H is said to be an M-cover of G if every vertex of G has exactly M copies in H. For an example of a graph cover, see Figure 1.

We will typically represent the hypergraph $G = (V, \mathcal{A})$ as a factor graph: a graph with a variable node for each vertex $i \in V$, a factor node for each $\alpha \in \mathcal{A}$, and an edge from $i \in V$ to $\alpha \in \mathcal{A}$ if and only if $i \in \alpha$. In this way, the above definition of graph covers can easily be extended to hypergraphs.

For a connected hypergraph $G = (V, \mathcal{A})$, each M-cover consists of a variable node for each of the $M|V|$ variables, a factor node for each of the $M|\mathcal{A}|$ factors, and an edge joining each distinct copy of $i \in V$ to a distinct copy of $\alpha \in \mathcal{A}$ whenever $i \in \alpha$.

To any M-cover $H = (V^H, \mathcal{A}^H)$ of G given by the homomorphism h, we can associate a collection of potentials: the potential at node $i \in V^H$ is equal to $\phi_{h(i)}$, the potential at node $h(i) \in G$, and for each $\alpha \in \mathcal{A}^H$, we associate the potential $\psi_{h(\alpha)}$. In this way, we can construct a function $f^H : \mathcal{X}^{M|V|} \to \mathbb{R}_{\geq 0}$ such that f^H factorizes over H. We will say that the graphical model $(H; \phi^H, \psi^H)$ is an M-cover of the graphical model $(G; \phi, \psi)$ whenever H is an M-cover of G and ϕ^H and ψ^H are derived from ϕ and ψ as above.

Finally, it will be convenient to express f^H as a function over M vectors in the set $\mathcal{X}^{|V|}$. We can partition the vertex set V^H into M disjoint sets $V_1, \ldots, V_M$ such that each set contains exactly one copy of each vertex in the graph G. Then, without loss of generality, any $x \in \mathcal{X}^{M|V|}$ can be expressed as $x^1, \ldots, x^m \in \mathcal{X}^{|V|}$ where x^m is an assignment to the variables in V_m for all $m \in \{1, \ldots, M\}$. In this case, we will write $f^H(x) = f^H(x^1, \ldots, x^M)$.

2.2 The Bethe Approximation

The Bethe free energy is a standard approximation to the so-called Gibbs free energy that is motivated by ideas from statistical physics. At temperature $T = 1$, the Bethe approximation, is defined as follows.

$$\begin{aligned}\log Z_{\mathrm{B}}(G,\tau;\phi,\psi) =& \sum_{i\in V}\sum_{x_i}\tau_i(x_i)\log\phi_i(x_i)\\ &+\sum_{\alpha\in\mathcal{A}}\sum_{x_\alpha}\tau_\alpha(x_\alpha)\log\psi_\alpha(x_\alpha)\\ &-\sum_{i\in V}\sum_{x_i}\tau_i(x_i)\log\tau_i(x_i)\\ &-\sum_{\alpha\in\mathcal{A}}\sum_{x_\alpha}\tau_\alpha(x_\alpha)\log\frac{\tau_\alpha(x_\alpha)}{\prod_{i\in\alpha}\tau_i(x_i)}\end{aligned}$$

for τ in the local marginal polytope,

$$\begin{aligned}\mathcal{T} \triangleq &\{\tau\geq 0 \mid \forall\alpha\in\mathcal{A}, i\in\alpha, \sum_{x_{\alpha\setminus i}}\tau_\alpha(x_\alpha)=\tau_i(x_i)\\ &\text{and } \forall i\in V, \sum_{x_i}\tau_i(x_i)=1\}.\end{aligned}$$

The Bethe partition function is defined to be the maximum value achieved by this approximation over $\mathcal{T}$.

$$Z_{\mathrm{B}}(G;\phi,\psi)=\max_{\tau\in\mathcal{T}} Z_{\mathrm{B}}(G,\tau;\phi,\psi)$$

The primary reason for the popularity of this approximation is that fixed points of the belief propagation algorithm correspond to stationary points of $\log Z_{\mathrm{B}}(G,\tau;\phi,\psi)$ over $\mathcal{T}$ (Yedidia et al., 2005). As such, all fixed points of the belief propagation algorithm provide a lower bound on the Bethe partition function. Our goal is to better understand the relationship between the Bethe partition function and the true partition function for different graphical models.

A recent theorem of Vontobel (2013) provides a combinatorial characterization of the Bethe partition function in terms of graph covers.

Theorem 2.2.

$$Z_{\mathrm{B}}(G;\phi,\psi)=\limsup_{M\to\infty}\sqrt[M]{\sum_{H\in\mathcal{C}^M(G)}\frac{Z(H;\phi^H,\psi^H)}{|\mathcal{C}^M(G)|}}$$

where $\mathcal{C}^M(G)$ is the set of all M-covers of G.

Proof. See Theorem 33 of (Vontobel, 2013). □

This characterization suggests that bounds on the partition functions of individual graph covers can be used to bound the Bethe partition function. This was the approach taken in (Ruozzi, 2012) and the approach that we will take in this work.

2.3 The Naïve Mean-Field Approximation

Another typical approximation, sometimes referred to as the naïve mean-field approximation, is to further restrict the local marginal polytope, the set $\mathcal{T}$ in the definition of the Bethe approximation, so that $\tau_\alpha(x_\alpha)=\prod_{i\in\alpha}\tau_i(x_i)$ for all $\alpha\in\mathcal{A}$.

As it is simply a specialization of Z_{B}, we must have that $Z_{\mathrm{MF}}(G;\phi,\psi)\leq Z_{\mathrm{B}}(G;\phi,\psi)$. However, this does not mean that the mean-field partition function is necessarily a worse approximation to the true partition function $Z(G;\phi,\psi)$, and unlike the Bethe partition function, the mean-field partition function always provides a lower bound on the true partition function (Jordan et al., 1999). More details about the mean-field approximation, its relationship to the Bethe approximation, and some experimental comparisons can be found in (Weiss, 2001).

3 Log-supermodularity and Lower Bounds

For a given graphical model $(G;\phi,\psi)$, we are interested in the relationship between the true partition function, $Z(G;\phi,\psi)$, and the Bethe approximation $Z_{\mathrm{B}}(G;\phi,\psi)$. In general, $Z_{\mathrm{B}}(G;\phi,\psi)$ can be either an upper or a lower bound on the true partition function, but for special families of graphical models, $Z_{\mathrm{B}}(G;\phi,\psi)$ is always a lower bound on $Z(G;\phi,\psi)$. In particular, this is true whenever the graphical model consists of only log-supermodular potentials.

Definition 3.1. *A function $f:\{0,1\}^n\to\mathbb{R}_{\geq 0}$ is called **supermodular** if for all $x,y\in\{0,1\}^n$*

$$f(x)+f(y)\leq f(x\wedge y)+f(x\vee y) \tag{1}$$

where $(x\wedge y)_i=\min\{x_i,y_i\}$ and $(x\vee y)_i=\max\{x_i,y_i\}$.

Definition 3.2. *A function $f:\{0,1\}^n\to\mathbb{R}_{\geq 0}$ is called **log-supermodular** if for all $x,y\in\{0,1\}^n$*

$$f(x)f(y)\leq f(x\wedge y)f(x\vee y). \tag{2}$$

Definition 3.3. *A graphical model $(G;\phi,\psi)$ is log-supermodular if for all $\alpha\in\mathcal{A}$, $\psi_\alpha(x_\alpha)$ is log-supermodular.*

The definition of submodular and log-submodular functions are obtained by reversing the inequality in (1) and (2) respectively. Log-supermodular functions have a number of special properties: the family is closed under multiplication and marginalization. In addition, they can be maximized in polynomial time. However, in general, computing (or even approximating) the partition function of a log-supermodular graphical model is computationally intractable (Goldberg and Jerrum, 2010).

For any collection of vectors $x^1, \ldots, x^M \in \{0,1\}^n$ let $x^{[1]}, \ldots, x^{[M]}$ denote the collection of vectors such that for all $i \in \{1, \ldots, n\}$ and all $m \in \{1, \ldots, M\}$, $x_i^{[m]}$ is the m^{th} largest element among $x_i^1, \ldots, x_i^M$. Equivalently, if $x^1, \ldots, x^M$ are the columns of some matrix B, then $x^{[1]}, \ldots, x^{[M]}$ are the columns of the matrix obtained from B by sorting the elements in each row from greatest to least.

In Ruozzi (2012), the following correlation inequality was proven for log-supermodular functions.

Theorem 3.4. *Let $f_1, \ldots, f_M : \{0,1\}^n \to \mathbb{R}_{\geq 0}$ and $g : \{0,1\}^{Mn} \to \mathbb{R}_{\geq 0}$ be nonnegative real-valued functions such that g is log-supermodular. If for all $x^1, \ldots, x^M \in \{0,1\}^n$,*

$$g(x^1, \ldots, x^M) \leq \prod_{m=1}^{M} f_m(x^{[m]}), \tag{3}$$

then

$$\sum_{x^1, \ldots, x^M \in \{0,1\}^n} g(x^1, \ldots, x^M) \leq \prod_{m=1}^{M} \Big[\sum_{x \in \{0,1\}^n} f_m(x) \Big].$$

Applying this theorem to log-supermodular factorizations, with a bit of rewriting, yields the following theorem.

Theorem 3.5. *If $(G; \phi, \psi)$ is a log-supermodular graphical model, then for any M-cover, $(H; \phi^H, \psi^H)$, of $(G; \phi, \psi)$, $Z(H; \phi^H, \psi^H) \leq Z(G; \phi, \psi)^M$.*

Corollary 3.6. *If $(G; \phi, \psi)$ is a log-supermodular graphical model, then*

$$Z_{\mathrm{MF}}(G; \phi, \psi) \leq Z_{\mathrm{B}}(G; \phi, \psi) \leq Z(G; \phi, \psi).$$

As the value of the Bethe approximation at any of the fixed points of BP is always a lower bound on $Z_{\mathrm{B}}(G; \phi, \psi)$, $Z_{\mathrm{B}}(G, \tau; \phi, \psi) \leq Z(G; \phi, \psi)$ for any fixed point of the BP algorithm, with corresponding beliefs τ, as well. Note however, that the inequality between $Z_{\mathrm{B}}(G, \tau; \phi, \psi)$ and the mean-field approximation is only guaranteed to hold for the optimal choice of τ in the local marginal polytope.

4 Pairwise Binary Models

In practice, many graphical models are not naturally formulated as log-supermodular models, but Theorem 3.5 is a statement only about log-supermodular functions. We want to understand when we can use Theorem 3.5 to obtain bounds on Z_{B} even when the graphical model is not log-supermodular. In this section, we motivate the results in this work by first examining this question for the special case of pairwise binary graphical models (i.e., $\mathcal{X} = \{0,1\}$ and $|\alpha| = 2$ for all $\alpha \in \mathcal{A}$). As each factor of a pairwise model depends exactly two variables, the hypergraph $G = (V, \mathcal{A})$ can be represented as a standard graph, and we will abuse notation and write $G = (V, E)$ in this case.

We can sometimes convert a graphical model that is not log-supermodular into a log-supermodular one by a change of variables (e.g., for a fixed $I \subseteq V$, a change of variables that sends $x_i \mapsto 1 - x_i$ for each $i \in I$ and $x_i \mapsto x_i$ for each $i \in V \backslash I$). These functions are the log-supermodular analog of the "switching supermodular" and "permuted submodular" functions considered in (Crama and Hammer, 2011) and (Schlesinger, 2007) respectively.

To illustrate this idea, consider the special case of log-submodular, bipartite graphical models. A graph $G = (V, E)$ is *bipartite* if the vertex set can be partitioned into two sets $A \subseteq V$ and $B = V \setminus A$ such that A and B are independent sets (i.e., there are no edges between any two vertices $v_1, v_2 \in A$ and similarly for B). We will denote bipartite graphs as $G = (A, B, E)$.

Let $G = (A, B, E)$ be a bipartite graph[1] and suppose that $(G; \phi, \psi)$ is a pairwise binary, log-submodular graphical model. For each edge $(a, b) \in E$ with $a \in A$ and $b \in B$ and all x_a, x_b, define $\psi'_{ab}(x_a, x_b) \triangleq \psi_{ab}(x_a, 1 - x_b)$. Similarly, for each $b \in B$ and all x_b, define $\phi'(x_b) \triangleq \phi(1 - x_b)$. Finally, for each $a \in A$ and all x_a, define $\phi'(x_a) \triangleq \phi(x_a)$. Observe that ψ'_{ab} is a log-supermodular function whenever ψ_{ab} is a log-submodular function. The new graphical model, $(G; \phi', \psi')$ is a pairwise binary, log-supermodular graphical model that is obtained from the original model via a change of variables. Consequently, both graphical models have the same partition functions and the same Bethe partition functions, and we can apply Theorem 3.5 to conclude that $Z(G; \phi, \psi) = Z(G; \phi', \psi') \geq Z_{\mathrm{B}}(G; \phi', \psi') = Z_{\mathrm{B}}(G; \phi, \psi)$.

5 The Potts and Random Cluster Models

As a first example of a family of graphical models with non-binary state spaces for which the Bethe partition function provides a lower bound on the true partition function, we consider the Potts model (with no external field) from statistical physics. For a fixed graph $G = (V, E)$, a positive integer q, and a vector of spins

[1]Factor graphs are always bipartite, but here we are requiring that the graph G is bipartite when represented as a standard graph.

$\sigma \in \{1, \ldots, q\}^{|V|}$,

$$f^G_{\text{Potts}}(\sigma; q, J) = \prod_{(i,j)\in E} e^{J_{ij}\delta(\sigma_i,\sigma_j)}$$

where each edge $(i, j) \in E$ is weighted by $J_{ij} \in \mathbb{R}$ and the notation emphasizes the dependence on the model parameters J and q. When $J_{ij} > 0$ for all $(i, j) \in E$, the model is said to be ferromagnetic (i.e., neighboring spins prefer to align), and the model is said to be antiferromagnetic if $J_{ij} < 0$ for all $(i, j) \in E$.

The case $q = 2$ corresponds to the Ising model. Each pairwise potential in the ferromagnetic Ising model is log-supermodular, so we can immediately apply Theorem 3.5 in order to show that Z_{B} gives a lower bound on the partition function. However, when $q > 2$ the pairwise potential functions need not be log-supermodular, even in the ferromagnetic case.

Our goal in this section is to show that, like switching log-supermodular functions, the ferromagnetic Potts model is also a log-supermodular function in disguise. Unlike the pairwise binary case, we will need more than variable switching in order to produce a log-supermodular function. First, we observe that the partition function of the ferromagnetic Potts model can be equivalently formulated as the partition function of a closely related model in statistical physics, the random cluster model.

The random cluster model is a measure over subsets of edges of the graph $G = (V, E)$ and is related to measures that arise in the study of percolation problems. For any subset $A \subseteq E$ and nonnegative weights p_{ij},

$$f^G_{\text{rc}}(A; q, p) = q^{k_G(A)} \prod_{(i,j)\in A} p_{ij}$$

where q is as above and $k_G(A)$ is the number of connected components of the graph $G = (V, A)$. If q is a positive integer, $J_{ij} \geq 0$ for all $(i, j) \in E$, and $p_{ij} = e^{J_{ij}} - 1$ for all $(i, j) \in E$, then $Z_{\text{rc}}(G; q, p) = Z_{\text{Potts}}(G; q, J)$. A short proof of this standard result from statistical physics can be found in Appendix A, and more details about the Potts model and its relationship to the random cluster model can be found in (Sokal, 2005) and (Grimmett, 2006).

5.1 Lower Bounds

As $k_G(A)$ is a supermodular function (here, think of A as being represented by its 0-1 indicator vector), $q^{k_G(A)}$ is a log-supermodular function. From this, we can conclude that f^G_{rc} is a log-supermodular function whenever $e^{J_{ij}} - 1 > 0$ for all $(i, j) \in E$.

Notice that f^G_{rc} does not factor over G as $q^{k_G(A)}$ depends on the entire set A. In fact, the factor graph corresponding to f^G_{rc} is a tree: it has one factor node that is adjacent to all of the variable nodes. Although, the Bethe free energy is exact for this model, computing the partition function remains computationally intractable. Instead, we will exploit the theoretical equivalence of the two partition functions to show that for any cover $(H; q, J^H)$ of the ferromagnetic Potts model $(G; q, J)$, $Z_{\text{Potts}}(H; q, J^H) \leq Z_{\text{Potts}}(G; q, J)^M$. Specifically, we will show how to apply Theorem 3.4 directly to the random cluster partition function corresponding to each cover of $(G; q, J)$ in order to show the desired inequality for the Potts model. See the end of Section 2.1 for a reminder of the notation related to graph covers.

Lemma 5.1. *Let $(H; q, J^H)$ be an M-cover of the Potts model $(G; q, J)$. For any $A = (A^1, \ldots, A^M) \subseteq E^H$,*

$$f^H_{\text{rc}}(A^1, \ldots, A^M; q, p^H) \leq \prod_{m=1}^{M} f^G_{\text{rc}}(A^{[m]}; q, p)$$

whenever $q \geq 1$ and $p_{ij} = e^{J_{ij}} - 1 \geq 0$ for all $(i, j) \in E^G$.

Proof. We will show that for any M-cover, H, of G,

$$k_H(A^1, \ldots, A^M) \leq \sum_{m=1}^{M} k_G(A^{[m]}).$$

The result will then follow from the definition of f_{rc} and the definition of H.

We prove this by induction on $|A|$, the number of edges in the set A. For the base case, $|A| = 0$, $k_H(\emptyset) = \sum_{m=1}^{M} k_G(\emptyset)$. Now, suppose $|A| > 0$. Fix one edge in $A = (A^1, \ldots, A^M)$ and let $B = (B^1, \ldots, B^M)$ be the edges of its corresponding connected component. There are two possibilities. First, if $B = A$, then the result follows from the observation that $\sum_{m=1}^{M} k_G(A^{[m]}) \geq \eta - 1 + k_H(A^1, \ldots, A^M)$ where η is the maximum element of the vector given by $\sum_{m=1}^{M} A^m$. Otherwise, if $B \neq A$, let $C = A \setminus B$. With the notation that $C = (C^1, \ldots, C^M)$,

$$\begin{aligned}
k_H(A) &= k_H(B) + k_H(C) - |V^H| \\
&\overset{(a)}{\leq} \sum_{m=1}^{M} k_G(B^{[m]}) + \sum_{m=1}^{M} k_G(C^{[m]}) - |V^H| \\
&\overset{(b)}{\leq} \sum_{m=1}^{M} k_G(A^{[m]}) + \sum_{m=1}^{M} k_G(\emptyset) - |V^H| \\
&= \sum_{m=1}^{M} k_G(A^{[m]})
\end{aligned}$$

where (a) follows from the induction hypothesis and (b) follows from the supermodularity of k_G. $\square$

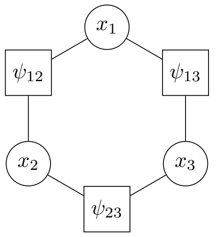


Figure 2: A simple cycle.

Theorem 5.2. *For the ferromagnetic Potts model with $q \geq 1$,*

$$Z_{\mathrm{MF}}(G; J, q) \leq Z_{\mathrm{B}}(G; J, q) \leq Z_{\mathrm{Potts}}(G; J, q).$$

Proof. By Lemma 5.1 and Theorem 3.4,

$$Z_{\mathrm{rc}}(H; q, p^H) \leq Z_{\mathrm{rc}}(G; q, p)^M$$

for any M-cover $(H; q, J^H)$ of $(G; q, J)$ with $p_{ij} = e^{J_{ij}} - 1$ for all $(i,j) \in E^G$ and $p^H_{ij} = e^{J^H_{ij}} - 1$ for all $(i,j) \in E^H$.

The proof then follows from the equivalence between Z_{Potts} and Z_{rc} and the characterization of Z_{B} in Theorem 2.2. $\square$

5.2 Potts Models with External Fields

Unlike the results of Ruozzi (2012) for the ferromagnetic Ising model, the results of Section 5.1 do not hold in the case that there is an arbitrary external field (i.e., arbitrary self-potentials). As an example, consider an instance of the ferromagnetic Potts model with an external field that factors over the graph in Figure 2.

$$f(x) = \prod_{k=1}^{3} e^{h^k_{x_k}} \prod_{i \neq j} e^{2\delta(x_i, x_j)}$$

where the vectors h^1, h^2, and h^3 are defined as follows.

$$h^1 = \begin{bmatrix} e^2 \\ e^{-1} \\ e^{-1} \end{bmatrix} \quad h^2 = \begin{bmatrix} e^{-1} \\ e^2 \\ e^{-1} \end{bmatrix} \quad h^3 = \begin{bmatrix} e^{-1} \\ e^{-1} \\ e^2 \end{bmatrix}$$

For this simple model, we can compute Z_{B} and Z exactly: $Z_{\mathrm{B}} - Z \approx 973.046$.

In general, the Potts model becomes much more difficult to analyze when there is an external field. However, for uniform external fields, we can extend the results of Section 5.1. Consider the following generalization of the Potts model for a given $h \in \mathbb{R}^q$.

$$f^G_{\mathrm{Potts+}}(\sigma; J, q, h) = \prod_{k \in V} e^{h_{\sigma_k}} \prod_{(i,j) \in E} e^{J_{ij}\delta(\sigma_i, \sigma_j)}$$

The corresponding random cluster representation is given by

$$f^G_{\mathrm{rc+}}(A; q, p, h) = \prod_{C \in \mathrm{comp}(A)} \Big[\sum_{w=1}^{q} e^{h_w |V(C)|}\Big] \prod_{(i,j) \in A} p_{ij}$$

where comp(A) is the set of connected components of the graph (V, A) and $V(C)$ is the set of vertices contained in the connected component C.

The equivalence between $Z_{\mathrm{Potts+}}$ and $Z_{\mathrm{rc+}}$ for appropriate choices of the parameters can be verified by the same techniques used in Appendix A. As before, $f^G_{\mathrm{rc+}}$ is a log-supermodular function of A, though the proof of this statement is non-trivial (see Theorem III.1 of (Biskup et al., 2000)). The analogs of Lemma 5.1 and Theorem 5.2 for this more general model can be proven by using the same arguments as before with minimal modification; we omit the proofs due to space constraints.

Lemma 5.3. *Let $(H; q, J^H, h^H)$ be an M-cover of the extended Potts model $(G; q, J, h)$. For any $A = (A^1, \ldots, A^M) \subseteq E^H$,*

$$f^H_{\mathrm{rc+}}(A^1, \ldots, A^M; q, p^H, h^H) \leq \prod_{m=1}^{M} f^G_{\mathrm{rc+}}(A^{[m]}; q, p, h)$$

whenever $q \geq 1$ and $p_{ij} = e^{J_{ij}} - 1 \geq 0$ for all $(i,j) \in E^G$.

Theorem 5.4. *For the extended ferromagnetic Potts model with $q \geq 1$,*

$$Z_{\mathrm{MF}}(G; J, q, h) \leq Z_{\mathrm{B}}(G; J, q, h) \leq Z_{\mathrm{Potts+}}(G; J, q, h)$$

5.3 Generalizations to Matroids

Theorem 5.2 and Lemma 5.1 can be extended, with arguments analogous to those in Appendix A, to Potts models defined on hypergraphs as considered in Goldberg and Jerrum (2010). The Potts and random cluster models (like the closely related Tutte polynomial) can also be generalized to matroids Sokal (2005). In this section, we describe the generalization to linear matroids and demonstrate the analog of Lemma 5.1 in this case.

Let S be a matrix over $GF(q)$ for some prime power q whose rows are indexed by the set V^S and whose columns are indexed by the set $\mathcal{A}^S$. The columns of S define a matroid with corresponding rank function $r_S(A)$ which is defined to be the rank, over $GF(q)$, of the submatrix formed by the columns indexed in $A \subseteq \mathcal{A}^S$. For $\sigma \in \{1, \ldots, q\}^{|V^S|}$, the normalized Potts model corresponding to this matroid is

$$f^S_{\mathrm{Potts}}(\sigma; q, J) = \frac{1}{q^{|V^S|}} \prod_{\alpha \in \mathcal{A}^S} \exp\Big[J_\alpha \delta(\sum_{i \in V^S} S_{i,\alpha}\sigma_i, 0)\Big]$$

where $\delta(\sum_i S_{i,\alpha}\sigma_i, 0) = 1$ if $\sum_i S_{i,\alpha}\sigma_i \equiv 0$ over $GF(q)$. Notice that the elements of S form the vertex-edge incidence matrix of a hypergraph $G^S = (V^S, \mathcal{A}^S)$. As a result, we will denote the graphical model for this generalization as $(S; q, J)$.

The partition function of the Potts model for the matroid over S can also be expressed as the partition function of an appropriate generalization of the random cluster model. For each $A \subseteq \mathcal{A}^S$, the normalized random cluster model over S is given by

$$f^S_{\text{rc}}(A; q, p) = q^{-r_S(A)} \prod_{\alpha \in A} p_\alpha.$$

The partition functions of these two models agree whenever $p_\alpha = \exp(J_\alpha) - 1$ for all $\alpha \in \mathcal{A}^S$. When $p_\alpha \geq 0$ for all $\alpha \in \mathcal{A}^S$, f^S_{rc} is a log-supermodular function. For a proof of the equivalence of $Z_{\text{rc}}(S; q, J)$ and $Z_{\text{Potts}}(S; q, J)$, see Theorem 3.1 of Sokal (2005).

Lemma 5.5. *For a prime power q and a matrix S over $GF(q)$, let $(S^H; q, J^H)$ be an M-cover of the normalized Potts model $(S; q, J)$.*

For any $A = (A^1, \dots, A^M) \subseteq \mathcal{A}^H$,

$$r_{S^H}(A^1, \dots, A^m) \geq \sum_{m=1}^{M} r_S(A^{[m]}).$$

Proof. As in the case of Lemma 5.1, the proof follows by induction on $|A| = |A^1| + \dots + |A^M|$. For $|A| = 0$ or 1, the result follows trivially. Suppose $|A| > 1$. We separate the proof into two cases. In the first case, $r_{S^H}(A) = |A|$. Fix any $a \in A$ and let $B = A \setminus \{a\}$ and $C = \{a\}$. We have

$$\begin{aligned} r_{S^H}(A^1, \dots, A^M) &= r_{S^H}(B) + r_{S^H}(C) \\ &\overset{(a)}{\geq} \sum_{m=1}^{M} r_S(B^{[m]}) + r_S(C) \\ &\overset{(b)}{\geq} \sum_{m=1}^{M} r_S(A^{[m]}) \end{aligned}$$

where (a) follows from the induction hypothesis and (b) follows from the submodularity of r_S.

For the second case, $r_{S^H}(A) < |A|$. Choose $B^1 \supseteq \ldots \supseteq B^M$ such that for all $m \in \{1, \dots, M\}$, $B^m \subseteq A^{[m]}$ and $|B^m| = r_S(A^{[m]})$. This can be accomplished by constructing a basis for the columns of S spanned by $A^{[M]}$ and then extending it to a basis of $A^{[M-1]}$ and so on. Similarly, construct an "unsorted" version of $B^1, \dots, B^M$ by choosing $C^1, \dots, C^M$ such that for all $m \in \{1, \dots, M\}$, $C^m \subseteq A^m$ and $C^{[m]} = B^{[m]}$. With these definitions, we have

$$\begin{aligned} \sum_{m=1}^{M} r_S(A^{[m]}) &= \sum_{m=1}^{M} r_S(B^m) \\ &\overset{(a)}{=} \sum_{m=1}^{M} r_S(C^m) \\ &\overset{(b)}{=} r_{S^H}(C^1, \dots, C^M) \\ &\overset{(c)}{\leq} r_{S^H}(A^1, \dots, A^M) \end{aligned}$$

where (a) follows from the submodularity of r_S, (b) follows from the induction hypothesis, and (c) follows from the monotonicity of r_{S^H}. □

With this lemma, we can again use the observation that Z_{Potts} and Z_{rc} are equivalent and apply Theorem 3.4 to conclude the following.

Theorem 5.6. *For any linear matroid S over $GF(q)$ for some prime power q,*

$$Z_{\text{Potts}}(S; J, q) \geq Z_{\text{B}}(S; J, q) \geq Z_{\text{MF}}(S; J, q)$$

whenever $J_\alpha \geq 0$ for all $\alpha \in \mathcal{A}^S$.

5.4 Weight Enumerators of Linear Codes

As an application of the results of Section 5.3, we demonstrate that the Bethe partition function can be used to compute lower bounds on the weight enumerator of linear codes over $GF(q)$ for some prime power q. Similar results have been demonstrated for the problem of counting the number of codewords of a binary cycle code (Vontobel, 2013). Let $S \in GF(q)^{k \times n}$ be the generator matrix of a linear code. A codeword corresponds to any vector in $GF(q)^n$ that is contained in the span of the rows of S. Denote the set of valid codewords as $\mathcal{C}$. The weight enumerator of a linear code is defined to be

$$\sum_{c \in \mathcal{C}} \lambda^{w(c)}$$

where λ is a positive real and $w(c)$ is the number of nonzero entries in the codeword c.

Equivalently, we can formulate this weight enumerator as the partition function of a generalized Potts model.

$$\begin{aligned} \sum_{c \in \mathcal{C}} \lambda^{w(c)} &= \sum_{\sigma \in \{1,\dots,q\}^k} \lambda^{w(\sum_{i=1}^{k} \sigma_i S_{i,*})} \\ &= \sum_{\sigma \in \{1,\dots,q\}^k} \prod_{\alpha \in \mathcal{A}^S} \lambda^{1-\delta(\sum_{i=1}^{k} \sigma_i S_{i,\alpha}, 0)} \\ &= \lambda^{|\mathcal{A}^S|} \sum_{\sigma \in \{1,\dots,q\}^{|V^S|}} \prod_{\alpha \in \mathcal{A}^S} \lambda^{-\delta(\sum_{i \in V^S} \sigma_i S_{i,\alpha}, 0)} \\ &= q^k \lambda^n Z_{\text{Potts}}(S; q, \log \frac{1}{\lambda}) \end{aligned}$$

where $S_{i,*}$ is the i^{th} row of S.

Using the previous results for the generalized ferromagnetic Potts model, we have that, for $\lambda \in (0,1]$,

$$\sum_{c\in\mathcal{C}} \lambda^{w(c)} \geq q^k \lambda^n Z_{\mathrm{B}}(S; q, \log\frac{1}{\lambda})$$
$$\geq q^k \lambda^n Z_{\mathrm{MF}}(S; q, \log\frac{1}{\lambda}).$$

As a consequence, belief propagation can be used to compute lower bounds on the weight enumerator in this regime.

6 Weighted Graph Homomorphisms

In this section, we consider the problem of counting weighted graph homomorphisms. For a fixed graph $G = (V, E)$, a nonegative matrix $\Gamma \in \mathbb{R}^{n\times n}$, and a vector of nonnegative weights $w \in \mathbb{R}^n$, consider the following function for each $\sigma \in \{1, \dots, n\}^{|V|}$.

$$f^G_{\mathrm{hom}}(\sigma; w, \Gamma) = \prod_{i\in V} w_{\sigma_i} \prod_{(i,j)\in E} \Gamma_{\sigma_i,\sigma_j}$$

If Γ is the adjacency matrix of a graph $\overline{G}$ and w is the all ones vector, then $Z_{\mathrm{hom}}(G; w, \Gamma)$ is equal to the number of graph homomorphisms from G to $\overline{G}$.

We will show that the Bethe partition function provides a lower bound on the true partition function, $Z_{\mathrm{hom}}(G; w, \Gamma)$, whenever

$$\Gamma = aa' + bb', \tag{4}$$

where a and b are two non-negative column vectors in $\mathbb{R}^n$ and v' denotes the transpose of the vector $v \in \mathbb{R}^n$.

For Γ as in (4), $Z_{\mathrm{hom}}(G; w, \Gamma)$ can be reformulated as the partition function of an edge coloring model. For each $A \subseteq E$, define

$$f^G_{\mathrm{edge}}(A; w, a, b) = \prod_{i\in V} \Big[\sum_{\sigma_i=1}^{n} w_{\sigma_i} a^{s_i(A)}_{\sigma_i} b^{|\partial i|-s_i(A)}_{\sigma_i}\Big]$$

where $s_i(A) = |\{j \in \partial i : (i,j) \in A\}|$. By convention, we take $0^0 = 1$.

The proof of equivalence between $Z_{\mathrm{hom}}(G; w, aa'+bb')$ and $Z_{\mathrm{edge}}(G; w, a, b)$ can be found in Appendix B. This proof is a special case of a more general relationship between the partition function of specific vertex coloring functions (our weighted homomorphism function) and the partition function of a more general class of edge coloring functions (Szegedy, 2007).

The edge coloring function f^G_{edge} is a log-supermodular function. However, unlike the ferromagnetic Potts model and its generalizations, the proof of this observation for the edge coloring model requires some work.

Lemma 6.1. *If a, b, and w are nonnegative vectors in $\mathbb{R}^n$, then $f^G_{\mathrm{edge}}(A; w, a, b)$ is a log-supermodular function of $A \subseteq E$.*

Proof. As a product of log-supermodualr functions is log-supermodular, it suffices to show that

$$\sum_{\sigma=1}^{n} w_\sigma a^{s_i(A)}_\sigma b^{|\partial i|-s_i(A)}_\sigma$$

is log-supermodular for each choice of $i \in V$ or equivalently that

$$\sum_{\sigma=1}^{n} x_\sigma c^{s_i(A)}_\sigma$$

is a log-supermodular function of A for any nonnegative vectors x and c.

Now, observe that, for all $A^1, A^2 \subseteq E$,

$$\Big[\sum_{\sigma=1}^{n} x_\sigma c^{s_i(A^1)}_\sigma\Big]\Big[\sum_{\sigma=1}^{n} x_\sigma c^{s_i(A^2)}_\sigma\Big] = \sum_{\sigma,\gamma} x_\sigma x_\gamma c^{s_i(A^1)}_\sigma c^{s_i(A^2)}_\gamma.$$

The desired inequality will follow from the observation that

$$c^{s_i(A^1)}_\sigma c^{s_i(A^2)}_\gamma + c^{s_i(A^2)}_\sigma c^{s_i(A^1)}_\gamma \leq$$
$$c^{s_i(A^1\vee A^2)}_\sigma c^{s_i(A^1\wedge A^2)}_\gamma + c^{s_i(A^1\wedge A^2)}_\sigma c^{s_i(A^1\vee A^2)}_\gamma \tag{5}$$

for all $\sigma, \gamma \in [1, \dots, n]$.

When $\sigma = \gamma$ in (5), the inequality is tight as $s_i(A^1 \wedge A^2) + s_i(A^1 \vee A^2) = s_i(A^1) + s_i(A^2)$. To show the inequality (5) when $\sigma \neq \gamma$, let $s = s_i(A^1 \wedge A^2) + s_i(A^1 \vee A^2)$, and observe that, as a function of $t \in \mathbb{R}$,

$$c^t_\sigma c^{s-t}_\gamma + c^{s-t}_\sigma c^t_\gamma$$

is symmetric about its minimum at $t = s/2$ and monotonic increasing for $t \geq s/2$. The inequality (5) follows from choosing $t = s_i(A^1 \vee A^2) \geq \max\{s_i(A^1), s_i(A^2)\} \geq s/2$.

□

Theorem 6.2. *For any graph $G = (V, E)$, any nonnegative $w \in \mathbb{R}^n_{\geq 0}$, and any matrix $\Gamma = aa' + bb'$ for some nonnegative vectors $a, b \in \mathbb{R}^n_{\geq 0}$,*

$$Z_{\mathrm{MF}}(G; w, \Gamma) \leq Z_{\mathrm{B}}(G; w, \Gamma) \leq Z_{homs}(G; w, \Gamma).$$

Proof. The factor graphs corresponding to the weighted homomorphism model and the edge coloring model are isomorphic (they simply exchange the role of the factor nodes and the variable nodes). As a result, every factor graph corresponding to an M-cover of a weighted homomorphism model is isomorphic to a factor graph of some M-cover of the edge coloring model. Because, f^G_{edge} is log-supermodular, an application of Theorem 3.5 gives that $Z_{\text{edge}}(G; w, a, b)^M \geq Z_{\text{edge}}(H; w^H, a^H, b^H)$ for any M-cover $(H; w^H, a^H, b^H)$ of $(G; w, a, b)$. The equivalence of the partition functions of the two models combined with the combinatorial characterization of the Bethe partition function in Theorem 2.2 then yields the desired result. □

7 Discussion

In a survey of submodular functions, Lovász (1983) noted that "submodularity is not a deep property, but it is often difficult to recognize the circumstances under which it occurs." The above examples serve to illustrate that the same is true of log-supermodularity.

For each model considered in this work, we transformed the problem of computing the partition function of a graphical model $(G; \phi, \psi)$ that is not log-supermodular into the computation of the partition function of a log-supermodular graphical model. This transformation was then applied to graph covers of $(G; \phi, \psi)$ in order to demonstrate that (3) holds for the transformed version of each cover. By the equality of the partition functions, this holds for the partition function of each graph cover of $(G; \phi, \psi)$ as well. Finally, we applied Theorem 3.4 to demonstrate that $Z_{\text{B}}(G; \phi, \psi) \leq Z(G; \phi, \psi)$. This implies that any fixed point of the belief propagation algorithm over the model $(G; \phi, \psi)$ must yield a lower bound on the true partition function.

Acknowledgments

This work was supported by EC grant FP7-265496, "STAMINA".

A The Potts and Random Cluster Models

In this appendix, we present a short algebraic proof of the equivalence between the partition function of the Potts model and the random cluster model under an appropriate choice of parameters. For any graph $G = (V, E)$ and any positive integer q, this equivalence follows primarily from the observation that $q^{k_G(A)}$ counts the number of colorings of the vertices of the graph G with q different colors such that any two vertices in the same connected component of the subgraph (V, A) are assigned the same color. This results in the following expression.

$$q^{k_G(A)} = \sum_{\sigma \in \{1,\ldots,q\}^{|V|}} \prod_{(i,j)\in A} \delta(\sigma_i, \sigma_j) \qquad (6)$$

for all $A \subseteq E$.

Given (6), the proof of equivalence follows from simple algebraic manipulations.

$$\begin{aligned}\sum_{A\subseteq E} f^G_{\text{rc}}(A; q, p) &= \sum_{A\subseteq E}\Big[q^{k_G(A)} \prod_{(i,j)\in A} p_{ij}\Big] \\ &= \sum_{A\subseteq E}\Big[\sum_{\sigma\in\{1,\ldots,q\}^{|V|}} \prod_{(i,j)\in A} p_{ij}\delta(\sigma_i,\sigma_j)\Big] \\ &= \sum_{\sigma\in\{1,\ldots,q\}^{|V|}} \Big[\sum_{A\subseteq E} \prod_{(i,j)\in A} p_{ij}\delta(\sigma_i,\sigma_j)\Big] \\ &= \sum_{\sigma\in\{1,\ldots,q\}^{|V|}} \prod_{(i,j)\in E} \Big[1 + p_{ij}\delta(\sigma_i,\sigma_j)\Big]\end{aligned}$$

Substituting $p_{ij} = e^{J_{ij}} - 1$ for all $(i, j) \in E$ yields

$$\begin{aligned}\sum_{A\subseteq E} f^G_{\text{rc}}(A; q, p) &= \sum_{\sigma\in\{1,\ldots,q\}^{|V|}} \prod_{(i,j)\in E} e^{J_{ij}\delta(\sigma_i,\sigma_j)} \\ &= \sum_{\sigma\in\{1,\ldots,q\}^{|V|}} f^G_{\text{Potts}}(\sigma; q, J).\end{aligned}$$

B The Weighted Homomorphism and Edge Coloring Models

In this appendix, we present a short algebraic proof of the equivalence between the partition function of the weighted homomorphism graphical model and the partition function of the edge coloring model whenever Γ is a rank two matrix of the form $aa' + bb'$ for two vectors a and b.

$$\begin{aligned}f^G_{\text{edge}}(A; a, b, w) &= \prod_{i\in V}\Big[\sum_{\sigma_i=1}^{n} w_{\sigma_i} a_{\sigma_i}^{s_i(A)} b_{\sigma_i}^{|\partial i| - s_i(A)}\Big] \\ &= \sum_{\sigma\in\{1,\ldots,n\}^{|V|}} \prod_{i\in V}\Big[w_{\sigma_i} a_{\sigma_i}^{s_i(A)} b_{\sigma_i}^{|\partial i| - s_i(A)}\Big] \\ &= \sum_{\sigma\in\{1,\ldots,n\}^{|V|}} \prod_{i\in V} w_{\sigma_i} \prod_{(i,j)\in E} c_{ij}(A)\end{aligned}$$

where

$$c_{ij}(A) = a_{\sigma_i}^{A_{(i,j)}} a_{\sigma_j}^{A_{(i,j)}} b_{\sigma_i}^{1-A_{(i,j)}} b_{\sigma_j}^{1-A_{(i,j)}}$$

and $A_{(i,j)} = 1$ if $(i, j) \in A$ and zero otherwise. Notice that $\sum_{A\subseteq E}\prod_{(i,j)\in E} c_{ij}(A) = \prod_{(i,j)\in E}\Gamma_{\sigma_i,\sigma_j}$ and that $c_{ij}(A)$ only depends on whether or not the edge $(i, j) \in A$. Consequently,

$$\begin{aligned}Z_{\text{edge}}(G; a, b, w) &= \sum_{A\subseteq E} f^G_{\text{edge}}(A; a, b, w) \\ &= \sum_{\sigma\in\{1,\ldots,n\}^{|V|}} \prod_{i\in V} w_{\sigma_i} \prod_{(i,j)\in E} \Gamma_{\sigma_i,\sigma_j} \\ &= Z_{\text{hom}}(G; w, \Gamma).\end{aligned}$$

References

N. Alon and J. H. Spencer. *The probabilistic method.* Wiley-Interscience series in discrete mathematics and optimization. Wiley, 2000.

M. Biskup, C. Borgs, J. T. Chayes, and R. Kotecky. Gibbs states of graphical representations of the Potts model with external fields. *Journal of Mathematical Physics*, 41(3):1170–1210, 2000.

Y. Crama and P. L Hammer. *Boolean functions: Theory, algorithms, and applications*, volume 142. Cambridge University Press, 2011.

L. Goldberg and M. Jerrum. Approximating the partition function of the ferromagnetic Potts model. In *Automata, Languages and Programming*, volume 6198 of *Lecture Notes in Computer Science*, pages 396–407. Springer Berlin Heidelberg, 2010.

G. Grimmett. *The Random-Cluster Model.* Number 333 in Grundl. Math. Wissen. Springer-Verlag, New York, 2006.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2): 183–233, 1999.

L. Lovász. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pages 235–257. Springer Berlin Heidelberg, 1983.

N. Ruozzi. The Bethe partition function of log-supermodular graphical models. In *Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, Dec. 2012.

Dmitrij Schlesinger. Exact solution of permuted submodular minsum problems. In *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, pages 28–38. Springer, 2007.

A. D. Sokal. *Surveys in Combinatorics 2005*, chapter The multivariate Tutte polynomial (alias Potts model) for graphs and matroids. Cambridge University Press, 2005.

B. Szegedy. Edge coloring models and reflection positivity. *J. Amer. Math. Soc.*, 20(4):969–988, 2007.

P. O. Vontobel. Counting in graph covers: A combinatorial characterization of the Bethe entropy function. *Information Theory, IEEE Transactions on*, Jan. 2013.

P. O. Vontobel and R. Koetter. Graph-cover decoding and finite-length analysis of message-passing iterative decoding of LDPC codes. *CoRR*, abs/cs/0512078, 2005.

Y. Weiss. *Advanced Mean Field Methods: Theory and Practice*, chapter Comparing the Mean Field Method and Belief Propagation for Approximate Inference in MRFs, pages 229 –239. MIT Press, 2001.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *Information Theory, IEEE Transactions on*, 51(7):2282 – 2312, July 2005.

Identifying Finite Mixtures of Nonparametric Product Distributions and Causal Inference of Confounders

Eleni Sgouritsa[1], **Dominik Janzing**[1], **Jonas Peters**[1,2], **Bernhard Schölkopf**[1]

[1] Max Planck Institute for Intelligent Systems, Tübingen, Germany
[2] Department of Mathematics, ETH, Zurich
{sgouritsa, janzing, peters, bs}@tuebingen.mpg.de

Abstract

We propose a kernel method to identify finite mixtures of nonparametric product distributions. It is based on a Hilbert space embedding of the joint distribution. The rank of the constructed tensor is equal to the number of mixture components. We present an algorithm to recover the components by partitioning the data points into clusters such that the variables are jointly conditionally independent given the cluster. This method can be used to identify finite confounders.

1 Introduction

Latent variable models are widely used to model heterogeneity in populations. In the following (Sections 2-4), we assume that the observed variables are jointly conditionally independent given the latent class. For example, given a medical syndrome, different symptoms might be conditionally independent. We consider $d \geq 2$ continuous observed random variables X_1, X_2,…,X_d with domains $\{\mathcal{X}_j\}_{1\leq j\leq d}$ and assume that their joint probability distribution $P(X_1, \dots, X_d)$ has a density with respect to the Lebesgue measure. We introduce a finite (i.e., that takes on values from a finite set) random variable Z that represents the latent class with values in $\{z^{(1)}, \dots, z^{(m)}\}$. Assuming $X_1, \dots, X_d$ to be jointly conditionally independent given Z (denoted by $X_1 \perp\!\!\!\perp X_2 \perp\!\!\!\perp \dots \perp\!\!\!\perp X_d \,|\, Z$) implies the following decomposition into a finite mixture of product distributions:

$$P(X_1, \dots, X_d) = \sum_{i=1}^{m} P(z^{(i)}) \prod_{j=1}^{d} P(X_j | z^{(i)}) \qquad (1)$$

where $P(z^{(i)}) = P(Z = z^{(i)}) \neq 0$.

By *parameter identifiability* of model (1), we refer to the question of when $P(X_1, \dots, X_d)$ uniquely determines the following parameters: (a) the number of mixture components m, and (b) the distribution of each component $P(X_1, \dots, X_d|z^{(i)})$ and the mixing weights $P(z^{(i)})$ up to permutations of z-values. In this paper, we focus on determining (a) and (b), when model (1) is identifiable. This can be further used to infer the existence of a hidden common cause (confounder) of a set of observed variables and reconstruct this confounder. The remainder of the paper is organized as follows: in Section 2, a method is proposed to determine (a), i.e., the number of mixture components m, Section 3 discusses established results on the identifiability of the parameters in (b). Section 4 presents an algorithm for determining these parameters and Section 5 uses the findings of the previous sections for confounder identification. Finally, the experiments are provided in Section 6.

2 Identifying the Number of Mixture Components

Various methods have been proposed in the literature to select the number of mixture components in a mixture model (e.g., Feng & McCulloch (1996); Böhning & Seidel (2003); Rasmussen (2000); Iwata et al. (2013)). However, they impose different kind of assumptions than the conditional independence assumption of model (1) (e.g., that the distributions of the components belong to a certain parametric family). Assuming model (1), Kasahara & Shimotsu (2010) proposed a nonparametric method that requires discretization of the observed variables and provides only a lower bound on m. In the following, we present a method to determine m in (1) without making parametric assumptions on the component distributions.

2.1 Hilbert Space Embedding of Distributions

Our method relies on representing $P(X_1, \dots, X_d)$ as a vector in a reproducing kernel Hilbert space (RKHS).

We briefly introduce this framework. For a random variable X with domain $\mathcal{X}$, an RKHS $\mathcal{H}$ on $\mathcal{X}$ with kernel k is a space of functions $f : \mathcal{X} \to \mathbb{R}$ with dot product $\langle \cdot, \cdot \rangle$, satisfying the reproducing property (Schölkopf & Smola, 2002): $\langle f(\cdot), k(x, \cdot)\rangle = f(x)$, and consequently, $\langle k(x, \cdot), k(x', \cdot)\rangle = k(x, x')$. The kernel thus defines a map $x \mapsto \phi(x) := k(x, .) \in \mathcal{H}$ satisfying $k(x, x') = \langle \phi(x), \phi(x')\rangle$, i.e., it corresponds to a dot product in $\mathcal{H}$.

Let $\mathcal{P}$ denote the set of probability distributions on $\mathcal{X}$, then we use the following *mean map* (Smola et al., 2007) to define a Hilbert space embedding of $\mathcal{P}$:

$$\mu : \mathcal{P} \to \mathcal{H}; \qquad P(X) \mapsto \mathbb{E}_X[\phi(X)] \tag{2}$$

We will henceforth assume this mapping to be injective, which is the case if k is *characteristic* (Fukumizu et al., 2008), as the widely used Gaussian RBF kernel $k(x, x') = \exp(-\|x - x'\|^2 / (2\sigma^2))$.

We use the above framework to define Hilbert space embeddings of distributions of every single X_j. To this end, we define kernels k_j for each X_j, with feature maps $x_j \mapsto \phi_j(x_j) = k(x_j, .) \in \mathcal{H}_j$. We thus obtain an embedding μ_j of the set $\mathcal{P}_j$ into $\mathcal{H}_j$ as in (2).

We can apply the same framework to embed the set of joint distributions $\mathcal{P}_{1,\dots,d}$ on $\mathcal{X}_1 \times \ldots \times \mathcal{X}_d$. We simply define a joint kernel $k_{1,\dots,d}$ by $k_{1,\dots,d}((x_1, \dots, x_d), (x'_1, \dots, x'_d)) = \prod_{j=1}^d k_j(x_j, x'_j)$, leading to a feature map into $\mathcal{H}_{1,\dots,d} := \bigotimes_{j=1}^d \mathcal{H}_j$ with $\phi_{1,\dots,d}(x_1, \dots, x_d) = \bigotimes_{j=1}^d \phi_j(x_j)$ (where $\bigotimes$ stands for the tensor product). We use the following mapping of the joint distribution:

$$\mu_{1,\dots,d} : \mathcal{P}_{1,\dots,d} \to \bigotimes_{j=1}^d \mathcal{H}_j \tag{3}$$

$$P(X_1, \dots, X_d) \mapsto \mathbb{E}_{X_1,\dots,X_d}[\bigotimes_{j=1}^d \phi_j(X_j)]$$

2.2 Identifying the Number of Components from the Rank of the Joint Embedding

By linearity of the maps $\mu_{1,\dots,d}$ and μ_j, the embedding of the joint distribution decomposes into

$$\begin{aligned}\mathcal{U}_{X_1,\dots,X_d} :=& \mu_{1,\dots,d}(P(X_1, \dots, X_d)) \\ =& \sum_{i=1}^m P(z^{(i)}) \bigotimes_{j=1}^d \mathbb{E}_{X_j}[\phi_j(X_j)|z^{(i)}] .\end{aligned} \tag{4}$$

Definition 1 (full rank conditional) *Let A, B two random variables with domains $\mathcal{A}, \mathcal{B}$, respectively. The conditional probability distribution $P(A|B)$ is called a full rank conditional if $\{P(A|b)\}_b$ with $b \in \mathcal{B}$ is a linearly independent set of distributions.*

Recalling that the rank of a tensor is the minimum number of rank 1 tensors needed to express it as a linear combination of them, we obtain:

Theorem 1 (number of mixture components) *If $P(X_1, \dots, X_d)$ is decomposable as in (1) and $P(X_j|Z)$ is a full rank conditional for all $1 \le j \le d$, then the tensor rank of $\mathcal{U}_{X_1,\dots,X_d}$ is m.*

Proof. From (4), the tensor rank of $\mathcal{U}_{X_1,\dots,X_d}$ is at most m. If the rank is $m' < m$, there exists another decomposition of $\mathcal{U}_{X_1,\dots,X_d}$ (apart from (4)) as $\sum_{i=1}^{m'} \bigotimes_{j=1}^d v_{i,j}$, with non-zero vectors $v_{i,j} \in \mathcal{H}_j$. Then, there exists a vector $w \in \mathcal{H}_1$, s.t. $w \perp \text{span}\{v_{1,1}, \dots, v_{m',1}\}$ and $w \not\perp \text{span}\{(\mathbb{E}_{X_1}[\phi_1(X_1)|z^{(i)}])_{1\le i \le m}\}$. The dual vector $,w\rangle$ defines a linear form $\mathcal{H}_1 \to \mathbb{R}$. By overloading notation, we consider it at the same time as a linear map $\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_d \to \mathcal{H}_2 \otimes \cdots \otimes \mathcal{H}_d$, by extending it with the identity map on $\mathcal{H}_2 \otimes \cdots \otimes \mathcal{H}_d$. Then, $\langle \sum_{i=1}^{m'} \bigotimes_{j=1}^d v_{i,j}, w\rangle = \sum_{i=1}^{m'} \langle v_{i1}, w\rangle \bigotimes_{j=2}^d v_{i,j} = 0$ but $\langle \mathcal{U}_{X_1,\dots,X_d}, w\rangle \ne 0$. So, $m = m'$. □

The assumption that $P(X_j|Z)$ is a full rank conditional, i.e., that $\{P(X_j|z^{(i)})\}_{i \le m}$ is a linearly independent set, is also used by Allman et al. (2009) (see Sec. 3). It does not prevent $P(X_j|z^{(q)})$ from being itself a mixture distribution, however, it implies that, for all j, q, $P(X_j|z^{(q)})$ is not a linear combination of $\{P(X_j|z^{(r)})\}_{r \ne q}$. Theorem 1 states that, under this assumption, the number of mixture components m of (1) (or equivalently the number of values of Z) is identifiable and equal to the rank of $\mathcal{U}_{X_1,\dots,X_d}$. A straightforward extension of Theorem 1 reads:

Lemma 1 (infinite Z) *If Z takes values from an infinite set, then the tensor rank of $\mathcal{U}_{X_1,\dots,X_d}$ is infinite.*

2.3 Empirical Estimation of the Tensor Rank from Finite Data

Given empirical data for every X_j, $\{x_j^{(1)}, x_j^{(2)}, \dots, x_j^{(n)}\}$, to estimate the rank of (4), we replace it with the empirical average

$$\hat{\mathcal{U}}_{X_1,\dots,X_d} := \frac{1}{n} \sum_{i=1}^n \bigotimes_{j=1}^d \phi_j(x_j^{(i)}) , \tag{5}$$

which is known to converge to the expectation in Hilbert space norm (Smola et al., 2007).

The vector (5) still lives in the infinite dimensional feature space $\mathcal{H}_{1,\dots,d}$, which is a space of functions

$\mathcal{X}_1 \times \cdots \times \mathcal{X}_d \to \mathbb{R}$. To obtain a vector in a finite dimensional space, we evaluate this function at the n^d data points $(x_1^{(q_1)}, \ldots, x_d^{(q_d)})$ with $q_j \in \{1, \ldots, n\}$ (the d-tuple of superscripts $(q_1, \ldots, q_d)$ runs over all elements of $\{1, \ldots, n\}^d$). Due to the reproducing kernel property, this is equivalent to computing the inner product with the images of these points under $\phi_{1,\ldots,d}$:

$$V_{q_1,\ldots,q_d} := \left\langle \hat{\mathcal{U}}_{X_1,\ldots,X_d}, \bigotimes_{j=1}^{d} \phi_j(x_j^{(q_j)}) \right\rangle$$
$$= \frac{1}{n} \sum_{i=1}^{n} \prod_{j=1}^{d} k_j(x_j^{(i)}, x_j^{(q_j)}) \qquad (6)$$

For $d = 2$, V is a matrix, so one can easily find low rank approximations via truncated Singular Value Decomposition (SVD) by dropping low SVs. For $d > 2$, finding a low-rank approximation of a tensor is an ill-posed problem (De Silva & Lim, 2008). By grouping the variables into two sets, say $X_1, \ldots, X_s$ and $X_{s+1}, \ldots, X_d$ without loss of generality, we can formally obtain the $d = 2$ case with two vector-valued variables. This amounts to reducing V in (6) to an $n \times n$ matrix by setting $q_1 = \cdots = q_s$ and $q_{s+1} = \cdots = q_d$. In theory, we expect the rank to be the same for all possible groupings. In practice, we report the rank estimation of the majority of all groupings. The computational complexity of this step is $O(2^{d-1}N^3)$.

3 Identifiability of Component Distributions and Mixing Weights

Once we have determined the number of mixture components m, we proceed to step (b) of recovering the distribution of each component $P(X_1, \ldots, X_d|z^{(i)})$ and the mixing weights $P(z^{(i)})$. In the following, we describe results from the literature on when these parameters are identifiable, for known m. Hall & Zhou (2003) proved that when $m = 2$, identifiability of parameters always holds in $d \geq 3$ dimensions. For $d = 2$ and $m = 2$ the parameters are generally not identifiable (there is a two-parameter continuum of solutions). In this case, one can obtain identifiability if, for all j, $P(X_j|Z)$ is pure (a conditional $P(X|Y)$ is called pure if for any two values y, y' of Y, the sum $P(X|y)\lambda + P(X|y')(1 - \lambda)$ is a probability distribution only for $\lambda \in [0, 1]$ (Janzing et al., 2011)). Allman et al. (2009) established identifiability of the parameters whenever $d \geq 3$ and for all m under weak conditions [1], using a theorem of Kruskal (1977). Finally, Kasahara & Shimotsu (2010) provided complementary identifiability results for $d \geq 3$ under different conditions with a constructive proof.

[1]the same assumption used in Theorem 1 that $P(X_j|Z)$ is a full rank conditional for all j.

4 Identifying Component Distributions and Mixing Weights

We are given n data points from an identifiable distribution $P(X_1, \ldots, X_d)$. Our goal is to cluster the data points (using m labels) in such a way that the distribution of points with label i is close to the (unobserved) empirical distribution of every mixture component, $P_n(X_1, \ldots, X_d|z^{(i)})$.

4.1 Existing Methods

Probabilistic mixture models or other clustering methods can be used to identify the mixture components (clusters) (e.g., Von Luxburg (2007); Böhning & Seidel (2003); Rasmussen (2000); Iwata et al. (2013)). However, they impose different kind of assumptions than the conditional independence assumption of model (1) (e.g., Gaussian mixture model). Assuming model (1), Levine et al. (2011) proposed an Expectation-Maximization (EM) algorithm for nonparametric estimation of the parameters in (1), given that m is known. Their algorithm uses a kernel as smoothing operator. They choose a common kernel bandwidth for all the components because otherwise their iterative algorithm is not guaranteed not to increase from one iteration to another. As stated also by Chauveau et al. (2010), the fact that they do not use an adaptive bandwidth (Benaglia et al., 2011) can be problematic especially when the distributions of the components differ significantly.

4.2 Proposed Method: Clustering with Independence Criterion (CLIC)

The proposed method, CLIC, assigns each of the n observations to one of the m (as estimated in Section 2) mixture components (clusters). We do not claim that each single data point is assigned correctly (especially when the clusters are overlapping). Instead, we aim to yield the variables jointly conditionally independent given the cluster in order to recover the components.

To measure conditional independence of $X_1, \ldots, X_d$ given the cluster we use the Hilbert Schmidt Independence Criterion (HSIC) (Gretton et al., 2008). It measures the Hilbert space distance between the kernel embeddings of the joint distribution of two (possibly multivariate) random variables and the product of their marginal distributions. If $d > 2$, we test for mutual independence. For that, we perform multiple tests, namely: X_1 against $(X_2, \ldots, X_d)$, then X_2 against $(X_3, \ldots, X_d)$ etc. and use Bonferroni correction. For each cluster, we consider as test statistic the HSIC from the test that leads to the smallest p-value ("highest" dependence).

We regard the negative sum of the logarithms of all p-values (each one corresponding to one cluster) under the null hypothesis of independence as our objective function. The proposed algorithm is iterative. We first randomly assign every data point to one mixture component. In every iteration we perform a greedy search: we randomly divide the data into disjoint sets of c points. Then, we select one of these sets and consider all possible assignments of the set's points to the m clusters (m^c possible assignments). The assignment that optimizes the objective function is accepted and the points of the set are assigned to their new clusters (which may coincide with the old ones). We, eventually, repeat the same procedure for all disjoint sets and this constitutes one iteration of our algorithm. After every iteration we test for conditional independence given the cluster. The algorithm stops after an iteration when any of the following happens: we observe independence in all clusters, no data point has changed cluster assignment, an upper limit of iterations is reached. It is clear that the objective function is monotonously decreasing.

The algorithm may not succeed at producing conditionally independent variables for different reasons: e.g., incorrect estimation of m from the previous step or convergence to a local optimum. In that case, CLIC reports that it was unable to find appropriate clusters.

Along the iterations, the kernel test of independence updates the bandwidth according to the data points belonging to the current cluster (in every dimension). Note, however, that this is not the case for the algorithm in Section 2. There, we are obliged to use a common bandwidth, because we do not have yet any information about the mixture components.

The parameter c allows for a trade-off between speed and avoiding local optima: for $c = n$, CLIC would find the global optimum after one step, but this would require checking m^n cluster assignments. On the other hand, $c = 1$ leads to a faster algorithm that may get stuck in local optima. In all experiments we used $c = 1$ since we did not encounter many problems with local optima. Considering c to be a constant, the computational complexity of CLIC is $O(m^c N^3)$ for every iteration. Algorithm 1 includes the pseudocode of CLIC.

Algorithm 1 CLIC

```
1: input data matrix $X$ of size $n \times d$, $m$, $c$
2: random assignment $cluster(i) \in \{1,\ldots,m\}, i = 1,\ldots,n$ of the data into $m$ clusters
3: while conditional dependence given cluster and clusters change do
4:    $obj = computeObj(cluster)$
5:    choose random partition $S_j, j = 1,\ldots,J$ of the data into sets of size $c$
6:    for $j = 1$ to $J$ do
7:       $newCluster = cluster$
8:       for all words $w \in \{1,\ldots,m\}^c$ do
9:          $newCluster(S_j) = w$
10:         $objNew(w) = computeObj(newCluster)$
11:      end for
12:      $wOpt = \text{argmin}(objNew)$
13:      $cluster(S_j) = wOpt$
14:   end for
15: end while
16: if conditional independence given cluster then
17:    output $cluster$
18: else
19:    output "Unable to find appropriate clusters."
20: end if
```

5 Causality: Identifying Confounders

Drawing causal conclusions from observed statistical dependences without being able to intervene on the system always relies on assumptions that link statistics to causality (Spirtes et al., 2001; Pearl, 2000). The least disputed one is the Causal Markov Condition (CMC) stating that every variable is conditionally independent of its non-descendant, given its parents (Spirtes et al., 2001; Pearl, 2000) with respect to a directed acyclic graph (DAG) that formalizes the causal relations. We focus on causal inference problems where the set of observed variables may not be causally sufficient, i.e. statistical dependences can also be due hidden common causes (confounders) of two or more observed variables. Under the assumption of linear relationships between variables and non-Gaussian distributions, confounders may be identified using Independent Component Analysis (Shimizu et al., 2009). Other results for the linear case are presented in Silva et al. (2006) and for the non-linear case with additive noise in Janzing et al. (2009). Fast Causal Inference (Silva et al., 2006) can exclude confounding for some pairs of variables, given that many variables are observed. Finally, the reconstruction of binary confounders under the assumption of pure conditionals is presented in Janzing et al. (2011).

In this section, we use the results of the previous sections to infer the existence and identify a finite confounder that explains all the dependences between the observed variables. We assume that latent variables are not caused by observed variables (the same assumption has been used by Silva et al. (2006)). Unlike previous methods, we do not make explicit assumptions on the distribution of the variables. Instead, we postulate a different assumption, namely that the conditional of each variable given its parents is full rank:

Assumption 1 (full rank given parents) *Let PA_Y denote the parents of a continuous variable Y with respect to the true causal DAG. Then, $P(Y|PA_Y)$ is a full rank (f.r.) conditional.*

Lemma 2 (full rank given parent) *By Assumption 1 it follows that, if $A \in PA_Y$ is one of the parents of Y, i.e., $A \to Y$, then, since $P(Y|PA_Y)$ is a f.r. conditional, $P(Y|A)$ is also a f.r. conditional (after marginalization).*

Remark: If Assumption 1 holds and $A \to B \to C$, then $P(B|A)$ and $P(C|B)$ are f.r. conditionals (Lemma 2), which implies that $P(C|A)$ is also a f.r. conditional, since it results from their multiplication.

Lemma 3 (shifted copies) *Let R be a probability distribution on $\mathbb{R}$ and $T_t R$ its copy shifted by $t \in \mathbb{R}$ to the right. Then $\{T_t R\}_{t\in\mathbb{R}}$ are linearly independent.*

Proof. Let

$$\sum_{j=1}^{q} \alpha_j T_{t_j} R = 0\,, \tag{7}$$

for some q and some q-tuple $\alpha_1, \ldots, \alpha_q$. Let $\hat{R}$ be the Fourier transform of R. If we set $g(\omega) := \sum_{j=1}^{q} \alpha_j e^{i\omega t_j}$ then (7) implies $g(\omega)\hat{R}(\omega) = 0$ for all $\omega \in \mathbb{R}$, hence g vanishes for all ω with $\hat{R}(\omega) \neq 0$, which is a set of non-zero measure. Since g is holomorphic, it therefore vanishes for all $\omega \in \mathbb{R}$ and thus all coefficients are zero. $\Box$

The following theorem shows that Assumption 1 is typically satisfied for a class of causal models considered by Hoyer et al. (2009):

Theorem 2 (additive noise model) *Let Y be given by $Y = f(PA_Y) + N$, where N is a noise variable that is statistically independent of PA_Y and f is an injective function. Then $P(Y|PA_Y)$ is a full rank conditional.*

The proof is a straightforward application of Lemma 3.

Theorem 3 (rank of cause-effect pair) *Assume there is a direct causal link $A \to B$ between two observed variables A and B, with $\{A,B\}$ not necessarily being a causally sufficient set (i.e., there may be unobserved common causes of A and B). Then, given Assumption 1, the rank of $\mathcal{U}_{A,B}$ is equal to the number of values that A takes, if A is finite. If A is infinite, then the rank of $\mathcal{U}_{A,B}$ is infinite.*

Proof. By Assumption 1, $P(B|A)$ is a f.r. conditional (Lemma 2). Since $A \perp\!\!\!\perp B \,|\, A$, applying Theorem 1 for finite $Z := A$ we conclude that the rank of $\mathcal{U}_{A,B}$ is equal to the number of values of A. For infinite A, we similarly apply Lemma 1 and we get infinite rank of $\mathcal{U}_{A,B}$. $\Box$

Theorem 4 (rank of confounded pair) *Assume $A \leftarrow C \to B$, and A, B do not have other common causes apart from C. Then, given Assumption 1, the rank of $\mathcal{U}_{A,B}$ is equal to the number of values of C.*

Proof. The proof is straightforward: by Assumption 1, $P(A|C)$ and $P(B|C)$ are f.r. conditionals (Lemma 2). Additionally, $A \perp\!\!\!\perp B \,|\, C$ and then, according to Theorem 1, the rank of $\mathcal{U}_{A,B}$ is equal to the number of values of $Z := C$ (for infinite C, the rank is infinite). $\Box$

Theorems 3 and 4 state what is the expected rank of $\mathcal{U}_{A,B}$ for various causal structure scenarios. However, in causal inference we are interested in inferring the *unknown* underlying causal DAG. The following Theorem uses Theorem 3 to infer the causal structure based on the rank of the embedding of the joint distribution.

Theorem 5 (identifying confounders) *Let $Y_1, \ldots, Y_d$ denote all observed variables. Assume they are continuous, pairwise dependent, and there is at most one (if any) hidden common cause of two or more of the observed variables. If Assumption 1 holds and the rank of $\mathcal{U}_{Y_1,\ldots,Y_d}$, with $d \geq 3$, is finite, then Fig. 1 depicts the only possible causal DAG with W being an unobserved variable and $P(Y_1, \ldots, Y_d, W)$ is identifiable up to reparameterizations of W.*

Proof. Assume there is at least one direct causal link $Y_i \to Y_{i'}$. Then, according to Theorem 3, the rank of $\mathcal{U}_{Y_i,Y_{i'}}$, and thus the rank of $\mathcal{U}_{Y_1,\ldots,Y_d}$, would be infinite. Therefore, direct causal links between the $\{Y_j\}$ can be excluded and the statistical dependencies between $\{Y_j\}$ can only be explained by hidden common causes. Since we assumed that there is at most one hidden common cause (and the observed variables are pairwise dependent), the only possible causal graph is depicted in Fig. 1. This implies $Y_1 \perp\!\!\!\perp Y_2 \perp\!\!\!\perp \ldots \perp\!\!\!\perp Y_d \,|\, W$ (according to the CMC), so model (1) holds, with $Z := W$ being the latent variable. According to the previous sections (Theorem 1 and Section 3), this model is identifiable. $\Box$

Based on Theorem 1, the number of values of W is equal to the rank of $\mathcal{U}_{Y_1,\ldots,Y_d}$ and $P(Y_1, \ldots, Y_d, W)$ can be identified according to Section 4. Note that the single common cause W could be the result of merging many common causes $W_1, .., W_k$ to one vector-valued variable W. Thus, at first glance, it seems that one does not lose generality by assuming only one common cause. However, Assumption 1, then, excludes the case where W consists of components each of which only acts on some different subset of the $\{Y_j\}$. $W_1, .., W_k$ should all be common causes of all $\{Y_j\}$.

$$\begin{tikzcd}
 & W \arrow[dl] \arrow[d] \arrow[drr] & & \\
Y_1 & Y_2 & \cdot\ \cdot & Y_d
\end{tikzcd}$$

Figure 1: Inferred causal DAG (the dotted circle represents an unobserved variable).

Note that, when we are given only finite data, the estimated rank of $\mathcal{U}_{Y_1,\ldots,Y_d}$ is always finite, highly depending on the strength of the causal arrows and the sample size. Then, we are faced with the issue that, based on Theorem 5, we would always infer that Fig. 1 depicts the only possible causal DAG, with the number of values of W being equal to the estimated rank. However, the lower the rank, the more confident we get that this is, indeed, due to the existence of a confounder that renders the observed variables conditionally independent (Fig. 1). On the other hand, high rank can also be due to direct causal links between the observed variables or continuous confounders. For that, we consider Theorem 5 to be more appropriate for inferring the existence of a confounder with a small number of values which would lead to low rank. However, we admit that what is considered "high" or "low" is not well defined. For example, how much "high" rank values we expect for the DAG $Y_1 \to Y_2$ highly depends on the strength of the causal arrow: the lower the dependence between Y_1 and Y_2 is, the lower is, generally, the estimated rank. In practice, we could make a vague suggestion that whenever the estimated rank is below 5 (although the dependence between $\{Y_j\}$ is strong), it is quite probable that this is due to a confounder (Fig. 1) but for higher rank it is getting more difficult to decide on the underlying causal structure.

6 Experiments

We conducted experiments both on simulated and real data. In all our experiments we use a Gaussian RBF kernel $k(x, x') = \exp\left(-\|x - x'\|^2/(2\sigma^2)\right)$. Concerning the first step of determining the number of mixture components: a common way to select the bandwidth σ_j for every k_j is to set it to the median distance between all data points in the jth dimension of the empirical data. However, this approach would usually result in an overestimation of the bandwidth, especially in case of many mixture components (see also Benaglia et al. (2011)). To partially account for this, we compute the bandwidth for every X_j as the median distance between points in the neighborhood of every point in the sample. The neighborhood is found by the 10 nearest neighbors of each point computed using all other variables apart from X_j. To estimate the rank of V, we find its SVD and report the estimated rank as $\hat{m} = \operatorname{argmin}_i(SV_{i+1}/SV_i)$ within the SVs that cover 90-99.999% of the total variance. Finally, concerning CLIC, we use 7 as the maximum number of iterations, but usually the algorithm terminated earlier.

6.1 Simulated Data

Simulated data were generated according to the DAG of Fig. 1 (we henceforth refer to them as the first set of simulated data), e.i., model (1) holds with $Z := W$, since $Y_1 \perp\!\!\!\perp Y_2 \perp\!\!\!\perp \ldots \perp\!\!\!\perp Y_d \,|\, W$. We first generated Z from a uniform distribution on m values. Then, the distribution of each mixture component in every dimension $(P(X_j|z^{(i)}))$ was chosen randomly between: (i) a normal distribution with standard deviation 0.7, 1, or 1.3, (ii) a t-distribution with degrees of freedom 3 or 10, (iii) a (stretched) beta distribution with alpha 0.5 or 1 and beta 0.5 or 1, and (iv) a mixture of two normal distributions with variance 0.7 for each. The distance between the components in each dimension was distributed according to a Gaussian with mean 2 and standard deviation 0.3. We chose the distance and the mixtures such that the experiments cover different levels of overlap between the components, and at the same time $\{P(X_j|z^{(i)})\}_{i\leq m}$ are generically linearly independent. We ran 100 experiments for each combination of $d = 2, 3, 5$ and $m = 2, 3, 4, 5$, with the sample size being $300 \times m$.

For comparison, we additionally generated data where the observed variables are connected also with direct causal links and thus are conditionally dependent given the confounder (we henceforth refer to them as the second set of simulated data). For that, we first generated data according to the DAG of Fig. 1, as above, for $d = 2$ and $m = 1$ (which amounts to no confounder) and for $d = 2$ and $m = 3$ (3-state confounder). X_2 was then shifted by $4X_1$ to simulate a direct causal link $X_1 \to X_2$. In this case, $X_1 \perp\!\!\!\perp X_2 \,|\, X_1$, so we have infinite $Z := X_1$.

6.1.1 Identifying the Number of Mixture Components

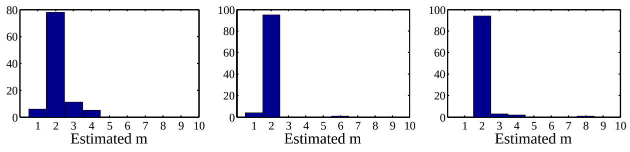


Figure 2: Histograms of the estimated number of mixture components m for the first set of simulated data, for $m = 2$ throughout, and $d = 2$ (left), $d = 3$ (middle), $d = 5$ (right).

We first report results on the first part of identifica-

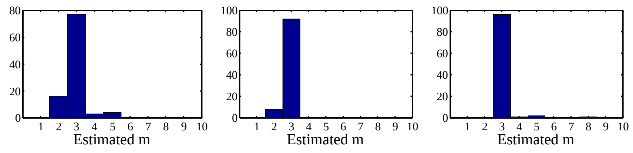


Figure 3: As Figure 2 but for $m = 3$.

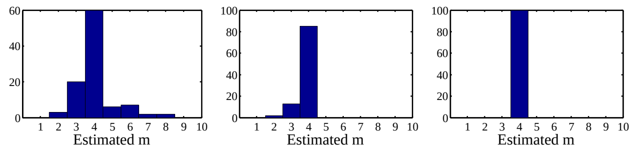


Figure 4: As Figure 2 but for $m = 4$.

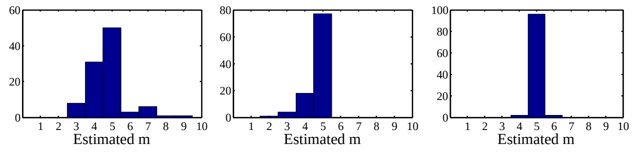


Figure 5: As Figure 2 but for $m = 5$.

tion, i.e. identifying the number of mixture components m in (1). The empirical rank estimation may depend on the strength of the causal arrows, the kernel bandwidth selection, the sample size and the way to estimate the rank by keeping only large eigenvalues. Figures 2, 3, 4 and 5 illustrate histograms of the estimated number of components (equivalently the estimated number of values of the confounder) for the first set of simulated data for $m = 2, 3, 4$ and 5, respectively. Each figure contains one histogram for every value of $d = 2, 3$ and 5. We can observe that as m increases the method becomes more sensitive in underestimating the number of components, a behavior which can be explained by the common sigma selection for all the data in each dimension or by high overlap of the distributions (which could violate Assumption 1). On the other hand, as d increases the method becomes more robust in estimating m correctly due to the grouping of variables that allows multiple rank estimations. The "low" estimated rank values provide us with some evidence that the causal DAG of Fig. 1 is true (Theorem 5). Of course, as stated also at the end of Sec. 5, it is difficult to define what is considered a low rank.

Figure 6 depicts histograms of the estimated number of components for the second set of simulated data. According to Theorem 3, the direct causal link $X_1 \to X_2$ results in an infinite rank of $\mathcal{U}_{X_1,X_2}$. Indeed, we can observe that in this case the estimated m is much higher. The "high" estimated rank values provide us with some evidence that the underlying causal DAG may include direct causal links between the observed variables or confounders with a high or infinite number of values. Note that, depending on the strength of the causal arrow $X_1 \to X_2$, we may get higher or lower rank values. For example, if the strength is very weak we get lower rank values since the dependence between X_1 and X_2 tends to be dominated by the confounder (that has a small number of values).

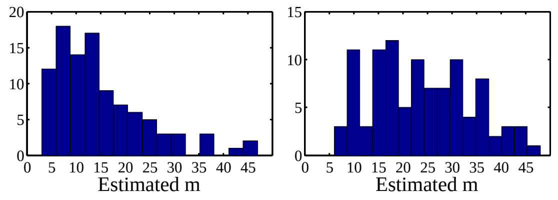


Figure 6: Histograms of the estimated m for the second set of simulated data (including a direct causal arrow). Left: no confounder, right: 3-state confounder.

6.1.2 Full Identification Framework

Next, we performed experiments, using the first set of simulated data to evaluate the performance of the proposed method (CLIC) (Section 4.2), the method of Levine et al. (2011) (Section 4.1) and the EM algorithm using a Gaussian mixture model (EM is repeated 5 times and the solution with the largest likelihood is reported). In the following, we refer to these methods as CLIC, Levine, and EM, respectively. For each data point, the two latter methods output posterior probabilities for the m clusters, which we sample from to obtain cluster assignments. Figure 7 illustrates the cluster assignments of these three methods for one simulated dataset with $m = 3$ and $d = 2$ (this is more for visualization purposes because for these values of d and m model (1) is not always identifiable). Note that permutations of the colors are due to the ambiguity of labels in the identification problem. However, EM incorrectly identifies a single component (having a mixture of two Gaussians as marginal density in X_1 dimension) as two distinct components. It is clear that this is because it assumes that the data are generated by a Gaussian mixture model and not by model (1), as opposed to CLIC and Levine methods.

We compare the distribution of each cluster output (for each of the three methods) to the empirical distribution, $P_n(X_1, \ldots, X_d|z^{(i)})$, of the corresponding mixture component (ground truth). For that we use the squared maximum mean discrepancy (MMD) (Gretton et al., 2012) that is the distance between Hilbert space embeddings of distributions. We only use the MMD and not one of the test statistics described in Gretton et al. (2012), since they are designed to compare two independent samples, whereas our samples

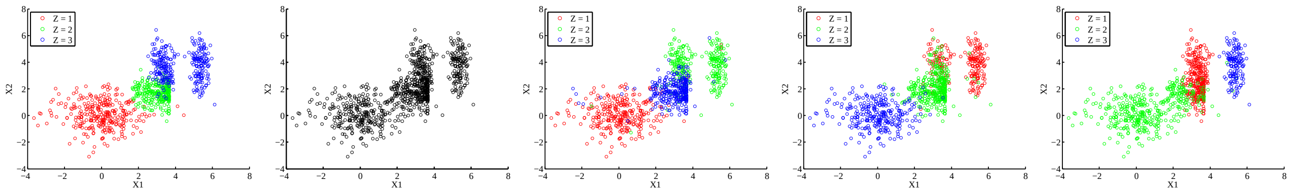


Figure 7: From left to right: ground truth, input, CLIC output, Levine output, and EM output for simulated data generated for $m = 3$. Each color represents one mixture component.

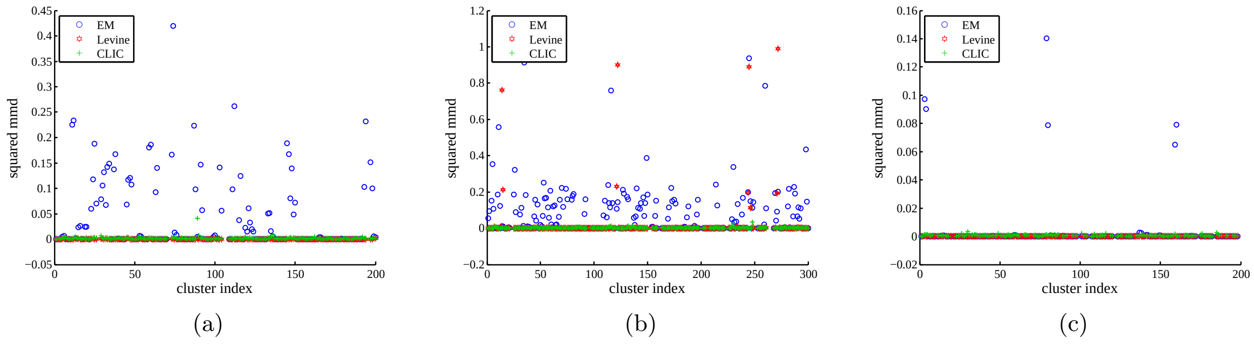


Figure 8: Squared MMD between output and ground truth clusters (for each of the three methods) for simulated data with (a) $d = 3, m = 2$, (b) $d = 3, m = 3$ and (c) $d = 5, m = 2$

(output and ground truth) have overlapping observations. To account for the permutations, we measure the MMD for all cluster permutations and select the one with the minimum sum of MMD for all clusters. Figures 8(a)-8(c) report the squared MMD results of the three methods for different combinations of m and d. Each point corresponds to the squared MMD for one cluster of one of the 100 experiments. Results are provided only for the cases that the number of components m was correctly identified from the previous step. The CLIC method was unable to find appropriate clusters in 2 experiments for $d = 3$ and $m = 3$ and in 13 for $d = 5$ and $m = 2$. Without claiming that the comparison is exhaustive, we can infer that both CLIC and Levine methods perform significantly better than EM, since they impose conditional independence. For higher d, EM improves since the clusters are less overlapping. However, the computational time of CLIC is higher compared to the other two methods.

6.2 Real data

Further, we applied our framework to the Breast Cancer Wisconsin (Diagnostic) Data Set from the UCI Machine Learning Repository (Frank & Asuncion, 2010). The dataset consists of 32 features of breast masses along with their classification as benign (B) or malignant (M). The sample size of the dataset is 569 (357 (B), 212 (M)). We selected 3 features, namely perimeter, compactness and texture, which are pairwise dependent (the minimum p-value is $pval = 2.43e - 17$), but become (close to) mutually independent when we condition on the class (B or M) ($pval_B = 0.016, pval_M = 0.013$). We applied our method to these three features (assuming the class is unknown) and we succeeded at correctly inferring that the number of mixture components is 2. Figure 9 depicts the ground truth of the breast data, the input and the results of CLIC, Levine and EM, and Fig. 12(a) the corresponding squared MMDs. We can observe that Levine method performs very poorly for this dataset.

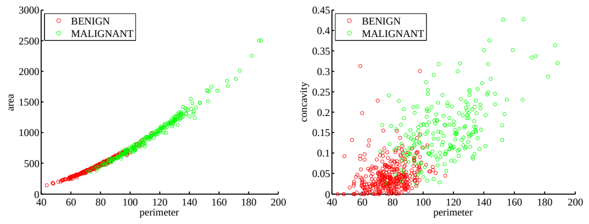


Figure 10: Breast data: features conditionally dependent given the class. Left: estimated $m = 62$, right: estimated $m = 8$.

Additionally, we selected different features, namely perimeter and area, and concavity and area, which are not conditionally independent given the binary class.

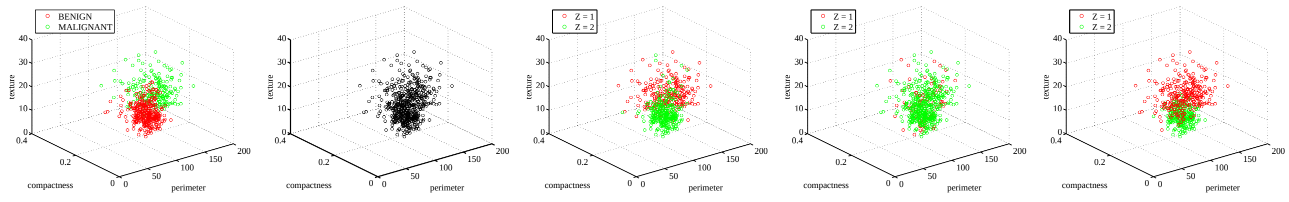


Figure 9: From left to right: ground truth, input, output CLIC, Levine, and EM for the breast data.

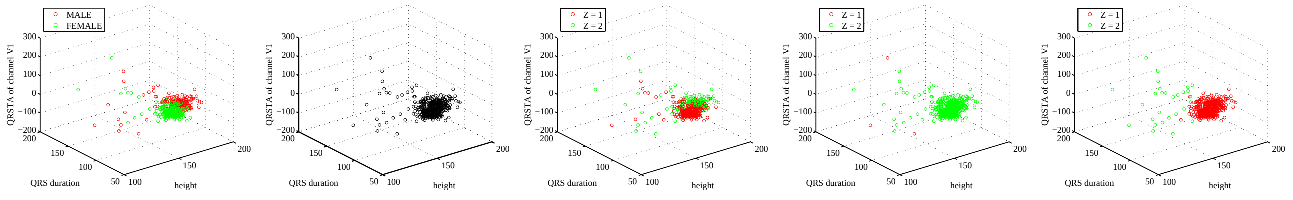


Figure 11: From left to right: ground truth, input, output CLIC, Levine, and EM for the arrhythmia data.

In this case, we got rank values higher than two, i.e., 62 and 8, respectively (Fig. 10).

We similarly applied our framework to the Arrhythmia Data Set (sample size 452) (Frank & Asuncion, 2010). We selected 3 features, namely height, QRS duration and QRSTA of channel V1 which are dependent (minimum $pval = 8.96e - 05$), but become independent when we condition on a fourth feature, the sex of a person (Male or Female) ($pval_M = 0.0607, pval_F = 0.0373$). We applied our method to the three features and we succeeded at correctly inferring that the number of mixture components is 2. Figure 11 depicts the ground truth, the input and the results of CLIC, Levine and EM, and Fig. 12(b) the corresponding squared MMDs. We can observe that Levine and EM methods perform very poorly for this dataset.

Finally, we applied our method to a database with cause-effect pairs[2]. It includes pairs of variables with known causal structure. Since there exists a direct causal arrow $X \to Y$, we expect the rank of $\mathcal{U}_{X,Y}$ to be infinite given our assumptions (Theorem 3), even if there exist hidden confounders or not. However, the estimated rank from finite data is always finite, its magnitude strongly depending on the strength of the causal arrow and the sample size, as mentioned in Sec. 5. Figure 13 depicts 4 cause-effect pairs (which were taken from the UCI Machine Learning Repository (Frank & Asuncion, 2010)) with the same sample size (1000 data points) but various degrees of dependence, specifically: $pval = 7.16e - 12$, $pval = 9.41e - 63$, $pval = 1.21e - 317$, $pval = 0$. The estimated ranks are $m = 1, 4, 8$ and 63, respectively. Note that when X and Y are close to independent (e.g., the first plot of Fig. 13) the assumption of pairwise dependence of Theorem 5 is almost violated.

[2]http://webdav.tuebingen.mpg.de/cause-effect/

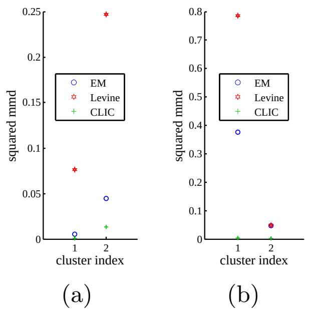


Figure 12: Squared MMD between output and ground truth clusters for (a) breast and (b) arrhythmia data.

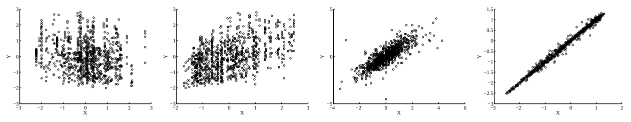


Figure 13: Four cause-effect pairs. Estimated m from left to right: $m = 1$, $m = 4$, $m = 8$, and $m = 63$.

7 Conclusion

In this paper, we introduced a kernel method to identify finite mixtures of nonparametric product distributions. The method was further used to infer the existence and identify a hidden common cause of a set of observed variables. Experiments on simulated and real data were performed for evaluation of the proposed approach. In practice, our method is more appropriate for the identification of a confounder with a small number of values.

References

Allman, E. S., Matias, C., and Rhodes, J. A. Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, 37:3099–3132, 2009.

Benaglia, M., Chauveau, D., and Hunter, D. R. Bandwidth selection in an EM-like algorithm for nonparametric multivariate mixtures. *In Nonparametrics and Mixture Models*, 2011.

Böhning, D. and Seidel, W. Editorial: recent developments in mixture models. *Computational Statistics & Data Analysis*, 41(3):349–357, 2003.

Chauveau, D., Hunter, D. R., and Levine, M. Estimation for conditional independence multivariate finite mixture models. *Technical Report*, 2010.

De Silva, V. and Lim, L.H. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1084–1127, 2008.

Feng, Z. D. and McCulloch, C. E. Using bootstrap likelihood ratios in finite mixture models. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(3):609–617, 1996.

Frank, A. and Asuncion, A. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Fukumizu, K., Gretton, A., Sun, X., and Schölkopf, B. Kernel measures of conditional dependence. In *Advances in Neural Information Processing Systems*, volume 20, pp. 489–496. MIT Press, 09 2008.

Gretton, A., Fukumizu, K., Teo, C. H., Song, L., Schölkopf, B., and Smola, A. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20*, pp. 585–592, Cambridge, 2008. MIT Press.

Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., and Smola, A. A kernel two-sample test. *Journal of Machine Learning Research*, 13:671 –721, 2012.

Hall, P. and Zhou, X.-H. Nonparametric estimation of component distributions in a multivariate mixture. *The Annals of Statistics*, 31(1):201–224, 2003.

Hoyer, P., Janzing, D., Mooij, J., Peters, J., and Schölkopf, B. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems 2008*. MIT Press, 2009.

Iwata, T., Duvenaud, D., and Ghahramani, Z. Warped mixtures for nonparametric cluster shapes. In *Uncertainty in Artificial Intelligence*, 2013.

Janzing, D., Peters, J., Mooij, J., and Schölkopf, B. Identifying latent confounders using additive noise models. In *Uncertainty in Artificial Intelligence*, pp. 249–257, 2009.

Janzing, D., Sgouritsa, E., Stegle, O., Peters, J., and Schölkopf, B. Detecting low-complexity unobserved causes. In *Uncertainty in Artificial Intelligence*, pp. 383–391, 2011.

Kasahara, H. and Shimotsu, K. Nonparametric identification of multivariate mixtures. Technical report, 2010.

Kruskal, J. B. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18(2):95 – 138, 1977.

Levine, M., Hunter, D. R., and Chauveau, D. Maximum smoothed likelihood for multivariate mixtures. *Biometrika*, 98(2):403–416, 2011.

Pearl, J. *Causality*. Cambridge University Press, 2000.

Rasmussen, Carl Edward. The infinite gaussian mixture model. *Advances in Neural Information Processing Systems*, 12(5.2):2, 2000.

Schölkopf, B. and Smola, A. *Learning with kernels*. MIT Press, Cambridge, MA, 2002.

Shimizu, S., Hoyer, P., and Hyvärinen, A. Estimation of linear non-gaussian acyclic models for latent factors. *Neurocomputing*, 72(7–9):2024–2027, 2009.

Silva, R., Scheines, R., Glymour, C., and Spirtes, P. Learning the structure of linear latent variable models. *JMLR*, 7:191–246, 2006.

Smola, A., Gretton, A., Song, L., and Schölkopf, B. A hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pp. 13–31. Springer-Verlag, 2007.

Spirtes, P., Glymour, C., and Scheines, R. *Causation, Prediction, and Search*. MIT Press, 2. edition, 2001.

Von Luxburg, U. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Determinantal Clustering Process - A Nonparametric Bayesian Approach to Kernel Based Semi-Supervised Clustering

Amar Shah
Department of Engineering
University of Cambridge
as793@cam.ac.uk

Zoubin Ghahramani
Department of Engineering
University of Cambridge
zoubin@eng.cam.ac.uk

Abstract

Semi-supervised clustering is the task of clustering data points into clusters where only a fraction of the points are labelled. The true number of clusters in the data is often unknown and most models require this parameter as an input. Dirichlet process mixture models are appealing as they can infer the number of clusters from the data. However, these models do not deal with high dimensional data well and can encounter difficulties in inference. We present a novel nonparameteric Bayesian method to cluster data points without the need to prespecify the number of clusters or to model complicated densities from which data points are assumed to be generated from. The key insight is to use determinants of submatrices of a kernel matrix as a measure of how close together a set of points are. We explore some theoretical properties of the model and derive a natural Gibbs based algorithm with MCMC hyperparameter learning. We test the model on various synthetic and real world data sets.

1 INTRODUCTION

Finding clusters amongst data points has been a key idea addressed by many researchers in machine learning, statistics and signal processing. In practice it is often the case that copious amounts of data can be easily collected, but subsequent labelling of the data is expensive and slow to obtain. *Semi-supervised learning* algorithms aim to utilise information from both labelled and unlabelled data to inform choices about how best to partition data points or where decision boundaries lie.

A natural approach for a Bayesian practitioner would be to consider a *generative model* of the data. This involves explicit modelling of the density which produced the observations and averaging over possible clusterings of the unlabelled training data. Next any parameters of the density model can be integrated out to produce a predictive clustering of unseen test data.

The choice of density model is integral and highly influential on the results of the training. A non-parametric Bayesian model is attractive in that it can incorporate an unbounded number of parameters and is able, in theory, to learn the correct density model e.g. Dirichlet Process Mixture Model [Escobar and West, 1995]. However such an approach can be expensive as typically, when clustering, we are not interested in the density itself whilst much effort is spent in learning it. Adams and Ghahramani [2009] propose a fully-Bayesian generative approach to semi-supervised classification which avoids the need to model complex density functions. Nonetheless this model does require prior specification of the number of classes and the training of a Gaussian process per class.

Discriminative models tend to be more popular for Bayesian semi-supervised learning [Zhu, 2005]. Lawrence and Jordan [2005] construct a nonparametric Bayesian model for binary semi-supervised classification, which is extended to the multi-class case by Rogers and Girolami [2007]. Similar Gaussian process based discriminative models that exploit graph-based information are suggested by Chu et al. [2007] and Sindhwani et al. [2007]. All of these examples also require knowledge of the total number of classes that the data is divided into.

In this work we present a novel discriminative nonparametric Bayesian method for clustering points using a kernel matrix determinant based measure of similarity between data points. It is nonparametric in that prior mass is assigned to all possible partitions of the data. This method is highly appropriate in the case where a generative model is computationally prohibitively expensive to train but where a kernel between pairs of data points can be easily computed e.g. in high dimensional data which cannot be adequately represented on a low dimensional manifold. Thus we bring together some of the most attracive properties of discriminative kernel methods (removing the need to model the input observations) and Bayesian nonparametrics (the ability to infer the number of clusters and kernel hyperparameters).

Our model makes use of a popular determinant based likelihood model called the *determinantal point process* (DPP) [Kulesza and Taskar, 2013]. This is a prior on the set of all subsets of a data set which places higher mass on subsets which contain diverse elements. DPPs arise in classical theory such as random matrix theory [Mehta and Gaudin, 1960, Ginibre, 1965] and quantum physics [Macchi, 1975]. They have more recently been used for human pose estimation, search diversification and document summarization [Kulesza and Taskar, 2010, 2011a,b].

In Section 2 we introduce the determinantal clustering process likelihood and discuss some of its properites. In Section 3 we explain how such a model can be applied to semi supervised clustering problems and develop a Gibbs based inference scheme with MCMC hyperparameter updates. We consider related research in Section 4 and describe the novel features of the DCP amongst other algorithms. Experimental performance of the model is outlined in Section 5 and finally conclusions are discussed in Section 6.

2 THE DETERMINANTAL CLUSTERING MODEL

Consider data points $X = \{x_n\}_{n=1}^N$ which live in a space $\mathcal{X}$. We assume that $x_i \neq x_j$ whenever $i \neq j$ without loss of generality (since we can assign any pair of points which violate this condition to the same cluster). In this work we typically consider the case $\mathcal{X} = \mathbb{R}^D$, but the ideas can easily be extended to more general spaces. Suppose further that we are given a positive definite kernel function, k, which depends on hyperparameters belonging to a space Θ, such that $k : \mathcal{X} \times \mathcal{X} \times \Theta \to \mathbb{R}$.

For a given set of hyperparameters, $\theta \in \Theta$ and ordered subsets $A, B \subseteq X$ of sizes N_A, N_B, define the $N_A \times N_B$ Gram matrix $K^\theta_{A,B}$ as

$$K^\theta_{A,B}(i,j) = k(x_{a_i}, x_{b_j}, \theta), \tag{1}$$

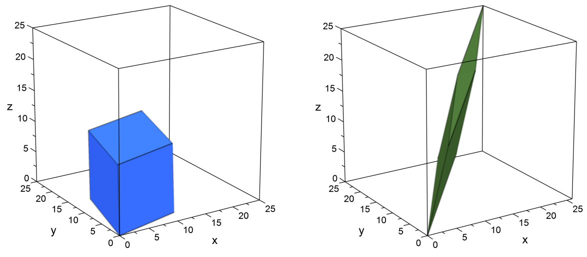
where x_{a_i} is the i^{th} element in A and x_{b_j} is the j^{th} element in B. For notational convenience we write $K^\theta_{A,A}$ as K^θ_A.

Let $\mathbb{S}$ be the set of all partitions of X. Hence an element $\mathcal{S} \in \mathbb{S}$ is a set of subsets of X such that for any $S, S' \in \mathcal{S}$, $S \cap S' = \emptyset$ and $\bigcup_{S\in\mathcal{S}} S = X$. We introduce the notion of a *determinantal clustering process* (DCP) for a given kernel function k and hyperparameters $\theta \in \Theta$, defined as a probability measure on the set $\mathbb{S}$ with density function

$$p(\mathcal{S} \in \mathbb{S}|\theta) \propto \prod_{S\in\mathcal{S}} \det\left(K^\theta_S\right)^{-1}. \tag{2}$$

Note that whilst DPPs have been extensively used as priors over subsets to encourage diversity [Kulesza and Taskar, 2013], we use the *inverse* of the determinant as a measure of similarity amongst points in a cluster which has very different properties which we discuss in the next section.

2.1 PROPERTIES OF THE DCP



(a) (b)

Figure 1: In (a) the volume represents the determinant of $\begin{pmatrix} 10 & 1 & 2 \\ 1 & 10 & 1 \\ 2 & 1 & 10 \end{pmatrix}$ whilst the in (b) the volume represents the determinant of $\begin{pmatrix} 10 & 8 & 7 \\ 8 & 10 & 8 \\ 7 & 8 & 10 \end{pmatrix}$. The more similar the rows of the matrix are, the smaller the determinant is.

Note that for any $S \in \mathcal{S} \in \mathbb{S}$ the matrix K^θ_S is positive definite implying that $\det(K^\theta_S) > 0$. The determinantal clustering process probability measure (Eq. (2)) therefore places positive mass on every $\mathcal{S} \in \mathbb{S}$. This is a crucial property which is highly attractive in a clustering task as it removes the need to prespecify the number of clusters the data should be partitioned into.

The geometric interpretation of the determinant of an $M \times M$ matrix is the *volume* spanned by its rows. A small determinant implies the rows are 'similar' to each other, whilst conversely a large determinant implies 'dissimilarity' amongst the rows. This phenomenon is illustrated in Figure 1. The reciprocal of the determinant is therefore a natural measure of similarity between data points.

Taking the product of such reciprocal determinants is a simple yet natural way to amalgamate the score for each cluster into a global score for a given partition. It also permits very simple sampling and inference which is discussed later.

Notice that unlike many popular clustering methods e.g. k-means or mixtures of Gaussians, the DCP does not assume that each cluster has a mean about which points occur with elliptically symmetrically. Such methods are often highly sensitive to initialisation and outliers which can be avoided by using a DCP type cluster scoring method. We explore this idea further in the Experiments section.

2.1.1 Relation to Gaussian Clustering in Feature Space

Suppose $\phi : \mathcal{X} \to \mathbb{R}^P$ is a non linear feature mapping for some $P \in \mathbb{N}$. Given data $x_1, ..., x_N \in \mathcal{X}$, let Φ be the $P \times N$ matrix with n^{th} column equal to $\phi(x_n)$. Now imagine that each row of Φ is an independent draw from a multivariate Gaussian with mean 0 and covariance Σ. The probability of Φ is given by

$$L(\Phi|\Sigma) \propto \frac{1}{\sqrt{\det(\Sigma)}} \exp\Big(-\frac{1}{2}\text{Trace}(\Phi\Sigma^{-1}\Phi^\top)\Big). \quad (3)$$

The maximum likelihood estimator of Σ is $\hat{\Sigma} = \Phi^\top\Phi$. Note that $\hat{\Sigma}$ is a kernel matrix with $\hat{\Sigma}_{m,n} = \phi(x_m)^\top\phi(x_n)$. Finally it is easy to show that

$$\begin{aligned} L(\Phi|\hat{\Sigma}) &\propto \frac{1}{\sqrt{\det(\Phi^\top\Phi)}} \exp\Big(-\frac{1}{2}\text{Trace}(\Phi(\Phi^\top\Phi)^{-1}\Phi^\top)\Big) \\ &\propto \det(\Phi^\top\Phi)^{-1/2}. \end{aligned} \quad (4)$$

The points $\phi(x_1), ..., \phi(x_N)$ are close together if and only if the covariance between points $\phi(x_1)_p, ..., \phi(x_N)_p$ is small for each $p \in \{1, ..., P\}$. This is the case exactly when $\det(\Phi^\top\Phi)$ is small. The probability of Φ is therefore large when $\phi(x_1), ..., \phi(x_N)$ are close together.

Suppose we partition the data and fit a Gaussian with the maximum likelihood covariance matrix to each partition. The resulting likelihood would be proportional to a product of terms like in Eq 4. This expression is identical to the DCP likelihood where we simply set $K = \Phi^\top\Phi$ and temper the likelihood (which is discussed in Section 2.2). The DCP does not require explicit specification of the feature map ϕ and works entirely on the kernel matrix. Note that clustering the columns of Φ leads to an entirely different process; Gaussian mixture modelling in feature space [Wang et al., 2003].

2.1.2 A Simple Example

Consider the delta kernel function

$$k(i,j,\theta) = \begin{cases} \theta & \text{if } x_i = x_j, \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Recall that $x_i \neq x_j$ for $i \neq j$, therefore for any $A \subseteq X$ of size N_A, $K_A^\theta = \theta I_{N_A}$ where I_{N_A} is an $N_A \times N_A$ identity matrix and $\det(K_A^\theta) = \theta^{N_A}$. In fact for any partition $\mathcal{S} \in \mathbb{S}$, $p(\mathcal{S} \in \mathbb{S}|\theta) \propto \theta^N$ which is independent of $\mathcal{S}$. For this choice of kernel, the DCP therefore places *uniform mass over the set of* all partitions. This is what we would expect intuitively since the kernel suggests that points are only similar to themselves and dissimilar to everything else.

2.1.3 Discouraging Singleton Clusters

A potential concern with such a model is that it may favour a large number of very small clusters. A similar problem was observed by Wu and Leahy [1993] when proposing a clustering scheme based on a minimum cut method, where leaf nodes tended to belong to their own clusters. We show that such a drawback does not exist under a DCP framework. A proof of the following can be found in Gentle [2007].

Lemma 1. *Suppose K, a symmetric positive definite matrix, is written in black matrix form,*

$$K = \begin{pmatrix} A & C^T \\ C & B \end{pmatrix},$$

then $\det(K) = \det(A)\det(B - CA^{-1}C^T)$.

For a kernel function k and hyperparameters $\theta \in \Theta$, consider a set $A \subset X$ and $x \in X \backslash A$. By applying Lemma 1, note that

$$\begin{aligned} &\det\big(K_{A\cup\{x\}}^\theta\big) \\ &\quad = \det\big(K_A^\theta\big)\det\big(K_{\{x\}}^\theta - K_{\{x\},A}^\theta {K_A^\theta}^{-1} K_{A,\{x\}}^\theta\big) \\ &\quad = \det\big(K_A^\theta\big) \times \big(K_{\{x\}}^\theta - K_{\{x\},A}^\theta {K_A^\theta}^{-1} K_{A,\{x\}}^\theta\big) \\ &\quad \leq \det\big(K_A^\theta\big) \times \big(K_{\{x\}}^\theta\big) \\ &\quad = \det\big(K_A^\theta\big)\det\big(K_{\{x\}}^\theta\big), \end{aligned}$$

where we use the fact that the determinant of a scalar is the scalar and the inequality comes from the fact that $y{K_A^\theta}^{-1}y^T > 0$ for any non-zero vector y by positive definiteness. We trivially deduce that

$$\det\big(K_{A\cup\{x\}}^\theta\big)^{-1} \geq \det\big(K_A^\theta\big)^{-1}\det\big(K_{\{x\}}^\theta\big)^{-1}$$

and hence that the DCP would always prefer to add a singleton to an existing cluster rather than to assign it to a new one.

It is important to appreciate that such a result does not necessarily hold when comparing the union of two sets each of size greater than 1 i.e. for $A, B \subset X$ disjoint each containing more than 1 element, it may be the case that

$$\det\left(K^{\theta}_{A\cup B}\right)^{-1} < \det\left(K^{\theta}_{A}\right)^{-1}\det\left(K^{\theta}_{B}\right)^{-1}.$$

If this were never possible the DCP would be a poor model as, we could show inductively, that its mode would be at the clustering where all points are clustered together for any set of data points X.

2.1.4 Choosing a Kernel Function

The behaviour of the DCP is entirely encoded in the functional form of the kernel and its parameters. This is entirely analogous to the fact that the behaviour of a support vector machine or a function drawn from a Gaussian process prior with mean 0 is entirely encoded in its covariance kernel function.

Many classes of positive definite kernel functions and more information about Gaussian processes can be found in Rasmussen and Williams [2006]. The squared exponential kernel is a common choice of kernel and is defined by

$$k(x,x',\{l,\sigma\}) = \sigma^2 \exp\Big(-\frac{1}{2}(x-x')^{\top}\mathrm{Diag}(l)^{-1}(x-x')\Big), \tag{6}$$

where $\mathrm{Diag}(l)$ is a diagonal matrix with l_i as the i^{th} diagonal entry. For any $N \times N$ positive definite matrix K, $\det(\alpha K) = \alpha^N \det(K)$. Since the DCP is a probability measure the constant multiplier becomes redundant hence we can set $\sigma = 1$ without losing any modelling flexibility.

2.2 Using a 'Temperature' Parameter

The addition of a *temperature* parameter to the DCP likelihood adds another layer of flexibility to the clustering process. Consider

$$p(\mathcal{S} \in \mathbb{S}|\theta) \propto \prod_{S\in\mathcal{S}} \det\left(K^{\theta}_{S}\right)^{-\tau}, \tag{7}$$

where $\tau \in \mathbb{R}_{+}$. This parameter has the effect of determining how peaked or flat the density is analogous to the temperature parameter in simulated annealing. Here, a large τ will make the density highly peaked at the mode whilst a small τ will encourage a uniform density over all partitions.

2.2.1 Kernel Hyperparameters

In some types of data analysis a user may actually know a good, application specific choice of kernel function and hyperparameters they wish to use. In such a case the DCP may be used as a prior over all possible clusterings directly with no further parameter learning required.

In most cases kernel hyperparameters are unknown apriori and have to be learned from data. We proceed under this assumption. In particular, we consider the case of observing many data points only some of which have been labelled. This is developed further in Section 3.1.2.

3 SEMI SUPERVISED CLUSTERING WITH DCP

Unlike for a classification problem, the 'name' of a particular cluster is irrelevant. For example, consider the clustering $\{\{1,4\},\{2,3\}\}$. Whether we call the set $\{1,4\}$ 'cluster 1' or 'cluster 2' is arbitrary, the important information is that 1 and 4 belong to the same cluster whilst 2 and 3 belong to another one.

We therefore can encode the relevant information about the clustering of points in X using a binary indicator matrix C, where for $x_i, x_j \in X$

$$C(x_i,x_j) = \begin{cases} 1 & \text{if } x_i,\, x_j \text{ in the same cluster,} \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Moreover notice that every partition $\mathcal{S} \in \mathbb{S}$ will have a unique such binary matrix representation which we denote $C_{\mathcal{S}}$.

In the semi-supervised setting we assume that some portion of our data set is labelled. Let $Z \subset X$ be the set of observed points for which we have labels, i.e. we observe the binary indicator matrix $\hat{C}_Z$ defined on pairs of inputs $x_i, x_j \in Z$.

For a given kernel function and hyperparameters $\theta \in \Theta$ our DCP likelihood function becomes

$$\begin{aligned} &p(\mathcal{S} \in \mathbb{S}|\hat{C}_Z,\theta,\tau) &(9)\\ &\quad\propto \prod_{S\in\mathcal{S}} \det\left(K^{\theta}_{S}\right)^{-\tau} \prod_{x,y\in Z} \mathbb{I}\big(C_{\mathcal{S}}(x,y) = \hat{C}_Z(x,y)\big), \end{aligned}$$

where $\mathbb{I}(.)$ is an indicator which takes value 1 when its argument is true and 0 otherwise. This second product encodes all the observed labels into the DCP model.

3.1 INFERENCE

Assuming a given parameterized kernel function, we describe a Gibbs based sampling method for allocating unlabelled data points to clusters and a MCMC step for learning kernel hyperparameters.

3.1.1 Sampling clusters

Suppose the sampler is currently at a particular partition $\mathcal{S} = \{S_1, ..., S_M\}$ for some integer $M \leq N$. Further suppose that for each cluster S_m, we have ${K^{\theta}_{S_m}}^{-1}$ stored in memory. We wish to update the cluster location of point $x \in X \backslash Z$ given the clustering of the remaining points. Without loss of generality, suppose $x \in S_M$ and let $\mathcal{S} \backslash \{x\} = \{S_1, ..., S_{M'}\}$ where $M' = M - 1$ if $S_M = \{x\}$ and $M' = M$ otherwise. In the latter case, we remove x from S_M and update ${K^{\theta}_{S_M}}^{-1}$ using the following lemma (proof found in Gentle [2007]).

Lemma 2. *Suppose we know K_A^{-1} for some non-empty set A. If we add an element $x \in X \backslash A$ to the set A, we have*

$$K^{-1}_{A \cup \{x\}} = \begin{pmatrix} U & V \\ V^T & \frac{1}{w} \end{pmatrix},$$

where

$$\begin{aligned} w &= K_{\{x\}} - K_{\{x\},A} K_A^{-1} K_{A,\{x\}}, \\ U &= K_A^{-1} + \frac{1}{w} K_A^{-1} K_{A,\{x\}} K_{\{x\},A} K_A^{-1}, \\ V &= -\frac{1}{w} K_A^{-1} K_{A,\{x\}}. \end{aligned}$$

We now must assign x to a particular cluster. For $m \in \{1, ..., M'\}$,

$$\begin{aligned} p(x \in S_m | \mathcal{S}\backslash\{x\}, \theta, \tau) &= \frac{p(\{S_1, .., S_m \cup \{x\}, .., S_{M'}\} | \theta, \tau)}{p(\{S_1, ..., S_{M'}\} | \theta, \tau)} \\ &\propto \frac{\det\left(K^{\theta}_{S_m \cup \{x\}}\right)^{-\tau}}{\det\left(K^{\theta}_{S_m}\right)^{-\tau}} \\ &\propto \left(K^{\theta}_{\{x\}} - K^{\theta}_{\{x\},S_m} {K^{\theta}_{S_m}}^{-1} K^{\theta}_{S_m,\{x\}}\right)^{-\tau}, \end{aligned} \tag{10}$$

and for $m = M' + 1$,

$$\begin{aligned} p(x \in S_m | \mathcal{S}\backslash\{x\}, \theta, \tau) &= \frac{p(\{S_1, ..., S_{M'}, \{x\}\} | \theta, \tau)}{p(\{S_1, ..., S_{M'}\} | \theta, \tau)} \\ &\propto \det\left(K^{\theta}_{\{x\}}\right)^{-\tau} \\ &\propto {K^{\theta}_{\{x\}}}^{-\tau}. \end{aligned} \tag{11}$$

We therefore allocate x to an existing cluster or a new cluster using a discrete uniform sample with these computed probabilities and update ${K^{\theta}_{S_m}}^{-1}$ using Lemma 2. This procedure is repeated for each $x' \in X \backslash Z$.

Note the conceptual similarity between this sampler and the collapsed Gibbs sampler for Dirichlet process mixtures; for each data point, the sampler decides whether to assign it to an existing (10) or new (11) cluster.

3.1.2 Sampling Kernel Hyperparameters and Temperature

Given a particular partition of the data $\mathcal{S} \in \mathbb{S}$, we wish to update the kernel hyperparameters and the temperature parameter using MCMC. We take $\psi = (\theta, \tau)$ to represent all these parameters in the proceeding discussion. Assume a prior density $p(\psi)$ over the parameter space $\Theta \times \mathbb{R}_+$. The posterior density for ψ is given by

$$p(\psi | \mathcal{S}) = \frac{p(\mathcal{S}|\psi) p(\psi)}{\int p(\mathcal{S}|\psi') p(\psi') d\psi'}. \tag{12}$$

We conjecture that the normalising constant of the DCP is analytically intractable. Whilst this is difficult to prove formally, we believe it to be true in part due to the sheer size of the set $\mathbb{S}$, known as the N^{th} *Bell number* [Wilf, 2006].

Consequently, we say that the posterior density is *doubly intractable* as the integral in the denominator is intractable and the likelihood in the numerator has an intractable normalising constant. A typical Metropolis-Hastings MCMC step would require the ability to compute the numerator of this posterior exactly.

To combat this issue we appeal to the Exchange Sampling algorithm of Murray et al. [2006] where we generate auxiliary data to avoid the need to compute normalising constants for the likelihood. Given that the current hyperparameters are set to $\psi \in \Theta \times \mathbb{R}_+$, suppose we have a proposal distribution $q(\psi \to \psi')$. The Single Variable Exchange Algorithm says to sample $\psi' \sim q(\psi \to \psi')$ and then to sample an auxiliary data set $\mathcal{S}' \sim p(\mathcal{S}'|\psi')$ (note that this can be done using the Gibbs based method of Section 3.1.1). The acceptance probability is set to

$$a = \min\left(1, \frac{q(\psi \to \psi') p(\mathcal{S}|\psi')}{q(\psi' \to \psi) p(\mathcal{S}|\psi)} \times \frac{p(\mathcal{S}'|\psi)}{p(\mathcal{S}'|\psi')}\right). \tag{13}$$

Notice the normalising constants for the observed data likelihoods cancel with those of the auxiliary data likelihoods removing the need to compute them explicitly. Murray et al. [2006] show that using such an acceptance probability, the Markov chain converges to the required posterior in the limit.

4 RELATED WORK

It is natural to question the relationship between determinantal clustering and spectral clustering [Shi and Malik, 2000]. Whilst both methods have similar matrix based inputs, their processes are fundamentally different. Spectral clustering maps this similarity matrix to its eigenspace and then performs a simple clustering algorithm e.g. k-means. Similarly kernel k-means [Dhillon et al.] maps the data to some feature space and performs k-means in this new space. In both examples, the kernel matrix is used to map inputs to a latent feature space before performing a simple clustering algorithm. This is not the case for the determinantal clustering process. The DCP simply uses the kernel to ensure positive definiteness so that determinants can be used as a measure of the size of a cluster. The DCP also does not require the prespecification of the number of clusters and learns this from the data.

The Dirichlet Process Gaussian Mixture Model (DPGMM) is a popular nonparametric Bayesian tool for clustering e.g. . This model assumes that each cluster is generated by an independent Gaussian distribution whose parameters are learnt from the data. Such a model requires modelling the joint distribution of all the data which can be difficult in high dimensions. Conversely the DCP requires just the ability to compute a kernel function between pairs of inputs.

There are plenty of flexible discriminative nonparameteric Bayesian models for multi-class classification problems based on transformed Gaussian processes [Lawrence and Jordan, 2005, Sindhwani et al., 2007, Chu et al., 2007, Rogers and Girolami, 2007, Adams and Ghahramani, 2009]. However, these all require knowledge of the number of classes apriori and are inappropriate for clustering tasks where the number of clusters is unknown.

Nonparametric clustering in spectral space is possible using the similarity-dependent Chinese restaurant process [Socher et al., 2011] or by combining the ideas of the DP-means algorithm [Kulis and Jordan, 2012] and kernel k-means to get a hard clustering which is able to infer the number of clusters.

5 EXPERIMENTS

We implement determinantal clustering on both synthetic and real data sets to demonstrate its properties. To provide a benchmark of results we compare the performance of the DCP with three other popular clustering methods: k-means, spectral clustering [Shi and Malik, 2000] and DPGMM [Escobar and West, 1995]. The DPGMM is a generative nonparametric Bayesian model whilst the other two methods require pre-specification of the number of clusters.

Two clustering metrics are computed to reflect the quality of clusterings sampled by these algorithms: adjusted rand index (ARI) Hubert and Arabie [1985] and normalized mutual information (NMI) [Manning et al., 2008]. Both are popular metrics for unsupervised clustering tasks. In cases where classes of data are actually known, we may use classification metrics such as precision and recall, however, we assume that we do not have this knowledge and only are interested in pairwise relationships between points. The ARI takes its maximum value at 1 for a perfect match in clustering, 0 represents a clustering which is equivalent in score to a random clustering and the ARI can also take negative values. The NMI is also maximized at 1 for a perfect clustering, but cannot take negative values. In our experiements, we compare these scores for the unlabelled test data points.

5.1 SYNTHETIC DATA

We illustrate two useful properties of the DCP over other clustering methods in this section. One common underlying assumption of clustering models is that data from a particular cluster are distributed elliptically symmetrically about some single cluster mean. Under this paradigm, the further away you are from a cluster mean, the less likely a point is to belong to that cluster. In many instances such an assumption is not a valid one and can lead to poor results. In the synthetic experiments we assume that the number of clusters is known. We use the squared exponential kernel for spectral and determinantal clustering learning the hyperparameters using cross validation and MCMC respectively. When computing clustering metrics for the DCP or DPGMM we sample partitions from the posterior and average scores over samples. For k-means and spectral clustering we average scores over a number of alternative initializations.

5.1.1 Clusters with Overlapping Boundaries

Consider the 2 cluster problem illustrated in Figure 2(a). Each cluster was generated from a two-dimensional Gaussian distribution and a few points were added near the boundary of the clusters which are not necessarily closer to their own cluster mean than the other cluster mean. In this experiment all the points other than the ones in squares were given labels and the task was to predict the cluster assignment of these points.

The performance of all models on the synthetic experiments is shown in Table 1. All models other than the DCP struggle with this type of data, especially k-means and spectral clustering since both procedures assign points to the clusters whose mean they are closest to. The DPGMM does slightly better as it allocates

Table 1: Results of Synthetic Experiments

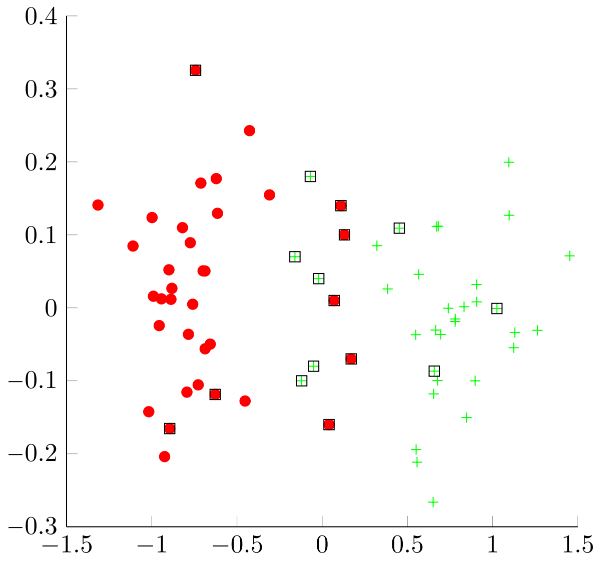
		DCP	DPGMM	k-means	Spectral
Overlap	ARI	**0.051**	0.006	-0.007	-0.004
	NMI	**0.143**	0.062	0.044	0.046
Multi Modal	ARI	**0.213**	-0.052	-0.127	-0.103
	NMI	**0.382**	0.176	0.150	0.122

points probabilistically, but the DCP is the outright winner. This is precisely due to the use of volume spanned by all points as opposed to distance from one point when determining clusters. Moving 1 point over a boundary may severely penalise the squared distance from the mean without affecting the cluster volume as adversarially. In this sense, the DCP is a more robust model.

5.1.2 Multi-Modal Clusters

Clusters of data points may actually be multimodal in their feature spaces. In such a case choosing a model which assumes elliptical symmetry about a single point is a poor choice. In Figure 2(b) we consider 2 clusters where the first is drawn from a mixture of two Gaussians and the second is drawn from a mixture of 3 Gaussians. Again, all points but the ones in black squares are labelled and the task was to predict the cluster assignments of these points.

Notice that an entire Gaussian mixture in the second cluster is unobserved. From the results in Table 1 we see that DPGMM, k-means and spectral clustering all perform poorly. Both DPGMM and k-means struggle because of their initialised cluster means. The hidden mixture component is roughly equidistant from these means, so during prediction roughly half of these points are assigned to one cluster and half to the other. Spectral clustering struggles as the learnt kernel parameters are essentially overfit to the training data. The performance of DCP is significantly better than these other models again reflecting the potential benefits of a volume based cluster size metric.


(a)

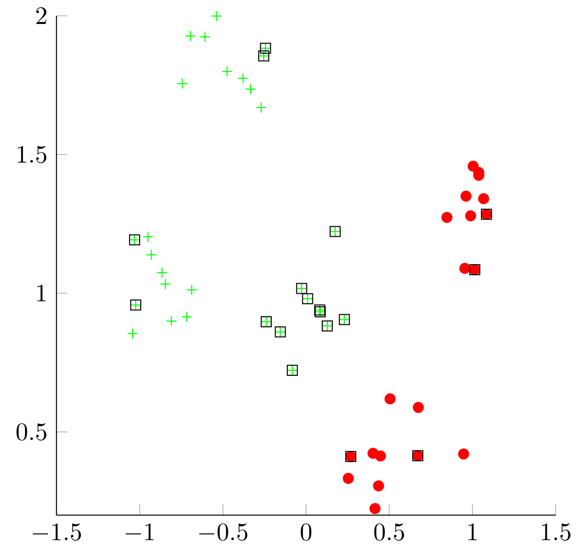

(b)

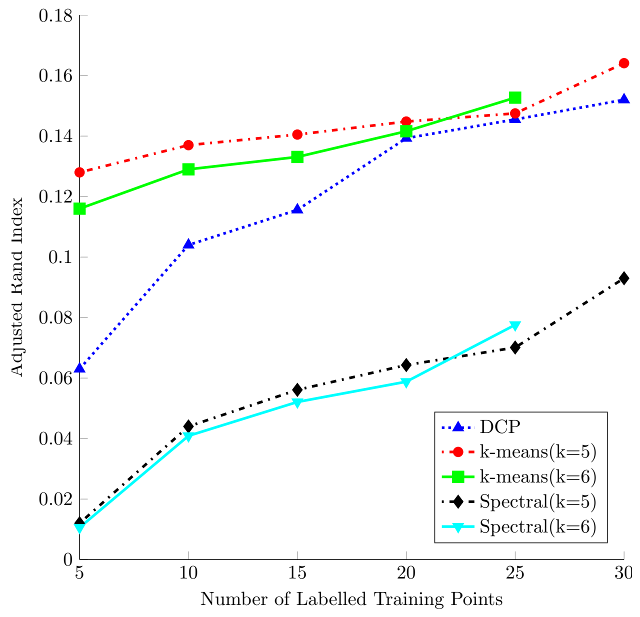
Figure 2: Synthetic datasets. Points to be predicted have dark squares.

5.2 REAL WORLD DATA

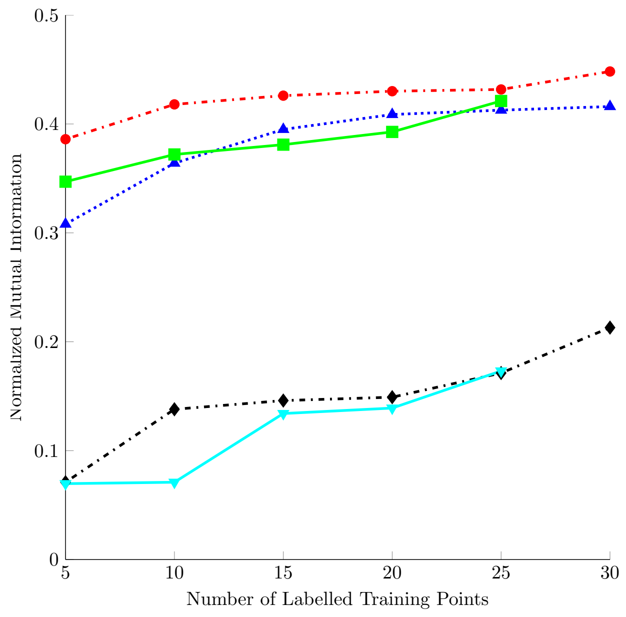
5.2.1 Wheat Kernels

This data set due to Charytanowicz et al. [2010] is a collection of geometric properties of 3 types of wheat kernels: Kama, Rosa and Canadian. The properties are real valued and include the wheat kernels' area, perimeter, compactness, length, width, asymmetry coefficient and length of kernel groove. We randomly select 20 examples from each type of wheat kernel to construct our data set. In this experiment we observe 6 labelled data points (3 from each of 2 randomly selected wheat types) and leave 5 data points from each wheat type as unobserved test points. The remaining 39 points are observed but unlabelled. The task was to predict the cluster assignments of the 15 test points. The results of the experiments are shown in Table 2.

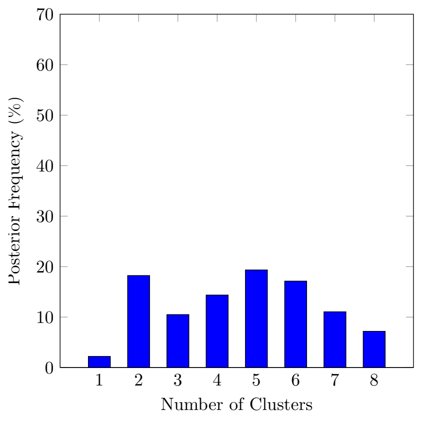
In this experiment the DPGMM outperforms other methods. Since the data is 7-dimensional, a mixture of Gaussians is still a powerful technique to use. For k-means and spectral clustering, the number of clusters has to be prespecified. Notice that there are only 2 clusters in the training data and that for $k = 2$ both models are poor choices. For $k = 3$, spectral clustering



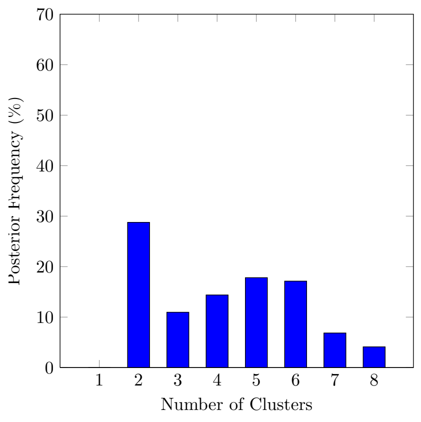
(a)



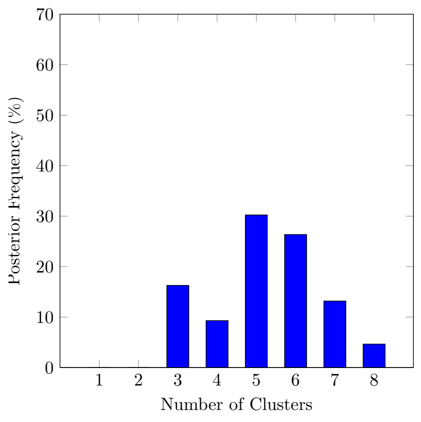
(b)



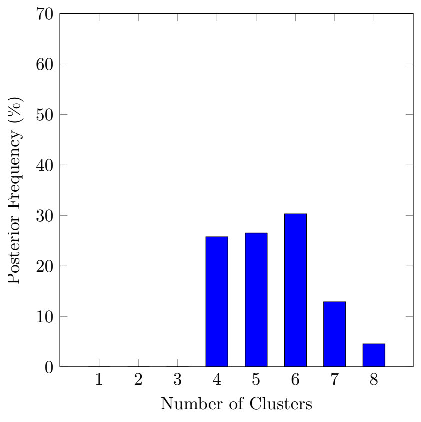
(c)



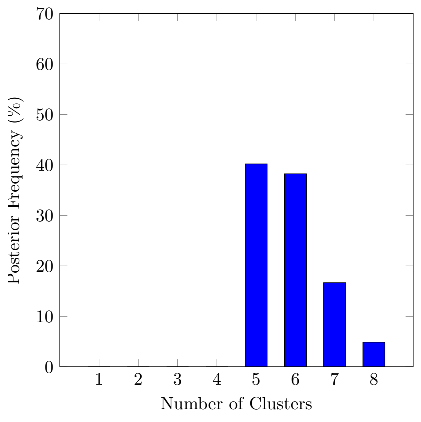
(d)



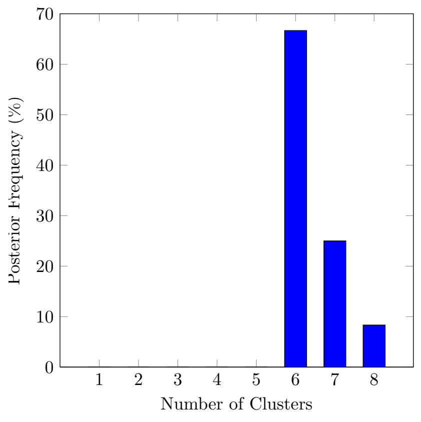
(e)



(f)



(g)



(h)

Figure 3: Results of the Car Lane Occupancy data experiments. We show (a) ARI and (b) NMI for each method varying the number of labelled training points. In (c)-(h) we plot the DCP posterior sampled number of clusters for 5, 10, 15, 20, 25 and 30 labelled points respectively.

Table 2: Results of Wheat Kernel Experiments

	DCP	DPGMM	k-means $k=2$	k-means $k=3$	Spectral $k=2$	Spectral $k=3$
ARI	0.696	**0.773**	0.355	0.513	0.397	0.707
NMI	0.767	**0.834**	0.510	0.608	0.527	0.778

does well, but only marginally better than the DCP.

5.2.2 Car Lane Occupancy Data

This data was collated by Cuturi [2011] from the Californian Department of Transportation PEMS website. The data describes occupancy rates between 0 and 1 of San Francisco bay area freeways every 10 minutes of every day for 15 months. A total of 963 road detectors were used. Hence for each day we have a $144 \times 963 = 138672$ long feature vector. The task is to cluster data from different days of the week together.

Since this data is extremely high dimensional, using a DPGMM is simply infeasible. In our experiments we extract data points every 2 hours rather than every 10 minutes, leaving our data 11566 dimensional and still beyond the capability of the DPGMM model. However, we are still able to compute a kernel between these feature vectors. In this experiment we consider a 1 parameter squared exponential kernel which has a shared lengthscale parameter across all dimensions.

First we select 6 days of the week: Saturday, Sunday and 4 weekdays. For each day we randomly select 20 data points and set aside 5 from each group as unseen test points; this gives 90 training and 30 test points. We vary the number of labelled points and try to predict the test points. In the i^{th} experiment we assume $5 \times i$ of the training points are labelled and belong in equal numbers to i different clusters. Therefore not only do we vary the number of labels, we also vary the number of observed cluster labels. The results are shown in Figure 3.

Spectral clustering results were generally poor here and this seems to be due to low flexibility of the kernel which only has 1 parameter. Despite the DCP using the same type of kernel function, it has significantly better results. This is due to the increased flexibility offered by the temperature parameter. The posterior sampler seemed to have a mode at around 4 for this data, which suggests that the 1 parameter kernel function was not sufficient in differentiating clusters.

We observe good performance from the k-means algorithm, in particular when we set k to the true value of 6. It should be noted that for $k = 5$ however, the DCP has competitive performance versus k-means as the number of labels increase, suggesting that when the number of clusters is truly unknown the DCP can be a powerful tool.

The sampled number of clustered under the DCP framework appears slightly multimodal at first. The peak at 2 is due to the process partitioning the weekend against the weekdays. This feature is most pronounced when labels from 2 clusters are observed in Figure 3(b) (one label was for a weekday the other for a weekend day). In Figure 3(f) there is no posterior mass on 1, 2 or 3 clusters and this is because labels for 4 clusters are observed so there is 0 likelihood of the data having less than 4 clusters.

6 CONCLUSIONS AND FUTURE WORK

In this work we have presented a novel kernel-based nonparameteric Bayesian approach to learning clusters in data. The key insight involves the use of kernel matrix determinants to score how close together subsets of data points are to each other. We discuss some elegant properties of the process and demonstrate its performance against other popular clustering methods.

Using a volume based cluster measurement proved beneficial for clusters which were not necessarily spread symmetrically about some mean point. A nonparametric Bayesian approach was shown to be fruitful when labelled data was scarce, whilst spectral clustering tended to overfit the kernel hyperparameters in cross validation, especially when the labelled data came from a small number of clusters in relation to the total number of clusters in the data set.

One drawback of such a model is that the computational cost of one cycle of Gibbs updates is $O(N^3)$ since each update requires matrix multiplication which is up to $O(N^2)$. An interesting research direction may be to use existing theory in matrix approximations to improve on this cost. Having a non-analytic normalising constant in the DCP likelihood adds another layer of difficulty in inference; exchange sampling is expensive. In future work it may be worth approximating this constant or using a variational method which makes hyperparameter learning relatively easy.

The remarkable property of this model is the fact that it overcomes the typically difficult task of dealing with complex high dimensional data. As long as any sensible positive definite kernel can be computed between pairs of data points, the determinantal clustering process can infer interesting properties about the data. This feature, combined with not having to prespecify the number of clusters makes the DCP a great contender for analysing complex data sets such as biological sequences, images, text and other multimedia.

References

R. P. Adams and Z. Ghahramani. Archipelago: Nonparametric Bayesian Semi-Supervised Learning. *Proceedings of the 26th International Conference on Machine Learning*, 2009.

M. Charytanowicz, J. Niewczas, P. Kulczycki, P. A. Kowalski, S. Lukasik, and S. Zak. *Information Technologies in Biomedicine*, volume 69 of *Advances in Intelligent and Soft Computing*. Springer Berlin Heidelberg, 2010.

W. Chu, V. Sindhwani, Z. Ghahramani, and S. S. Keerthi. Relational learning with Gaussian Processes. *Advances in Neural Information Processing Systems*, pages 289–296, 2007.

M. Cuturi. Fast global alignment kernels. *Proceedings of the 28th International Conference on Machine Learning*, pages 929–936, 2011.

I. Dhillon, Y. Guan, and B. Kulis.

M. D. Escobar and M. West. Bayesian Density Estimation and Inference Using Mixtures. *Journal of the American Statistical Association*, 90:577–588, 1995.

J. E. Gentle. *Matrix Algebra: Theory, Computations and Applications in Statistics*. Springer, 2007.

J. Ginibre. Statistical Ensembles of Complex, Quaternion and Real Matrices. *Journal of Mathematical Physics*, 6:440, 1965.

L. Hubert and P. Arabie. Comparing Partitions. *Journal of Classification*, pages 193–218, 1985.

A. Kulesza and B. Taskar. Structured Determinanteal Point Processes. *Advances in Neural Information Processing Systems*, 2010.

A. Kulesza and B. Taskar. Learning Determinantal Point Processes. *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, 2011a.

A. Kulesza and B. Taskar. k-dpps: Fixed-size Determinantal Point Processes. *Proceedings of the 28th International Conference on Machine Learning*, 2011b.

A. Kulesza and B. Taskar. Determinantal Point Processes for Machine Learning, 2013. URL `http://arxiv.org/abs/1207.6083`.

B. Kulis and M.I. Jordan. Revisiting k-means: New Algorithms via Bayesian Nonparametrics. *Proceedings of the 29th International Conference on Machine Learning*, 2012.

N. D. Lawrence and M. I. Jordan. Semi-supervised Learning via Gaussian Processes. *Advances in Neural Information Processing Systems*, pages 753–760, 2005.

O. Macchi. The Coincidence Approach to Stochastic Point Processes. *Advances in Applied Probability*, 7(1):83–122, 1975.

C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

M. L. Mehta and M. Gaudin. On the density of Eigenvaluse of a Random Matrix. *Nuclear Physics*, 18(0): 420–427, 1960.

I. Murray, Z. Ghahramani, and D. MacKay. Mcmc for Doubly-Intractable Distributions. *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 359–366, 2006.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

S. Rogers and M. Girolami. Multi-class Semi-supervised Learning with the ϵ-truncated Multinomial Probit Gaussian Process. *Journal of Machine Learning Research: Gaussian Processes in Practise*, pages 17–32, 2007.

J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on PAMI*, 2000.

V. Sindhwani, W. Chu, and S. S. Keerthi. Semi-supervised Gaussian process classifiers. *International Joint Conference on Artificial Intelligence*, pages 1059–1064, 2007.

R. Socher, A. Maas, and C. D. Manning. Spectral Chinese Restaurant Processes: Nonparametric Clustering Based on Similarities. *Proceedings of the 14th Conference on Artificial Intelligence and Statistics*, 2011.

J. Wang, J. Lee, and C. Zhang. Kernel Trick Embedded Gaussian Mixture Model. *Proceedings of the 14th International Conference of Algorithmic Learning Theory*, pages 159–174, 2003.

H. S. Wilf. *Generatingfunctionology*. A K Peters, Ltd., 3rd edition, 2006.

Z. Wu and R. Leahy. An Optimal Graph Theoretic Approach to Data Clustering: Theory and its Application to Image Segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15:11:101–113, 1993.

X. Zhu. Semi-Supervised Learning Literature Survey. *Technical Report 1530, Dept. of Computer Sciences, University of Wisconsin, Madison*, 2005.

Sparse Nested Markov Models with Log-linear Parameters

Ilya Shpitser
Mathematical Sciences
University of Southampton
i.shpitser@soton.ac.uk

Robin J. Evans
Statistical Laboratory
Cambridge University
rje42@cam.ac.uk

Thomas S. Richardson
Department of Statistics
University of Washington
thomasr@u.washington.edu

James M. Robins
Department of Epidemiology
Harvard University
robins@hsph.harvard.edu

Abstract

Hidden variables are ubiquitous in practical data analysis, and therefore modeling marginal densities and doing inference with the resulting models is an important problem in statistics, machine learning, and causal inference. Recently, a new type of graphical model, called the nested Markov model, was developed which captures equality constraints found in marginals of directed acyclic graph (DAG) models. Some of these constraints, such as the so called 'Verma constraint', strictly generalize conditional independence. To make modeling and inference with nested Markov models practical, it is necessary to limit the number of parameters in the model, while still correctly capturing the constraints in the marginal of a DAG model. Placing such limits is similar in spirit to sparsity methods for undirected graphical models, and regression models. In this paper, we give a log-linear parameterization which allows sparse modeling with nested Markov models. We illustrate the advantages of this parameterization with a simulation study.

1 Introduction

Analysis of complex multidimensional data is often made difficult by the twin problems of hidden variables, and a dearth of data relative to the dimension of the model. The former problem motivates the study of marginal and/or latent models, while the latter has resulted in the development of sparsity methods.

A particularly appealing model for multidimensional data analysis is the Bayesian network or directed acyclic graph (DAG) model [10], where random variables are represented as vertices in the graph, with directed edges (arrows) between them. The popularity of DAG models stems from their well understood theory, and from the fact that they elicit an intuitive causal interpretation: an arrow from a variable A to a variable B in a DAG model can be interpreted, in a way which can be made precise, to mean that A is a 'direct cause' of B.

DAG models assume all variables are observed, and a latent variable model based on DAGs simply relaxes this assumption. However, latent variables introduce a number of problems: it is difficult to correctly model the latent state, and the resulting marginal densities are quite challenging to work with. An alternative is to encode constraints found in marginals of DAG models directly; a recent approach in this spirit is the nested Markov model [15]. The advantage of the nested Markov model is that it correctly captures the conditional independences and other equality constraints found in marginals of DAG models. However, the discrete parameterization of nested Markov models has the disadvantage of being unable to represent constraints in various marginals of DAGs *concisely*, that is with few non-zero parameters. This implies that model selection methods based on scoring (via the BIC score [13] for instance) often prefer simpler models which fail to capture independences correctly, but which contain many fewer parameters [15].

More generally, in high dimensional data analyses there is often such a shortage of samples that classical statistical inference techniques do not work. To address these issues, sparsity methods have been developed, which drive as many parameters in the statistical model to zero as possible, while still providing a reasonable fit to the data. Sparsity methods have been developed for regression models [16], undirected graphical models [8, 9], and even some marginal models [4].

It is not natural to apply sparsity techniques to existing parameterizations of nested Markov models, because the parameters are context (or strata) specific.

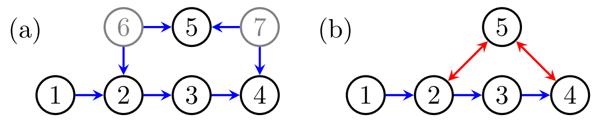


Figure 1: (a) A DAG with nodes 6 and 7 representing hidden variables. (b) An ADMG representing the same conditional independences as (a) among the variables corresponding to $1, 2, 3, 4, 5$.

In this paper, we develop a log-linear parameterization for discrete nested Markov models, where the parameters represent (generalizations of) log odds-ratios within 'kernels' (informally 'interventional' densities). These can be viewed as interaction parameters, of the kind commonly set to zero by sparsity methods. Our parameterization allows us to represent distributions containing 'Verma constraints' in a sparse way, while maintaining advantages of nested Markov models, and avoiding the disadvantages of using marginals of DAG models directly.

2 Disadvantages of the Möbius Parameterization of Nested Markov Models

One drawback of the standard parameterization of nested Markov models is that parameters are variation dependent; that is, fixing the value of one parameter constrains the 'legal' values of other parameters. This is in direct contrast with parameterizations of DAG models where parameters associated with a particular Markov factor (a conditional density for a variable given all its parents in the DAG) do not depend on parameters associated with other Markov factors.

We illustrate another difficulty with an example. Here, and in subsequent discussions, we will need to draw distinctions between vertices in graphs, and corresponding random variables in distributions or 'kernels.' We use the following notation: v (lowercase) denotes a vertex, X_v the corresponding random variable, and x_v a value assignment to this variable. Likewise A (uppercase) denotes a vertex set, X_A the corresponding random variable set, and x_A an assignment to this set.

Consider the marginal DAG shown in Fig. 1 (a). We wish to avoid representing this domain with a DAG directly, in order not to commit to a particular state space of the unobserved variables X_6 and X_7, and because, even if we were willing to make such an assumption, the margin over $(X_1, X_2, X_3, X_4, X_5)$ obtained from a density that factorizes according to this DAG can be complicated to work with [7].

To use nested Markov models for this domain, we first construct an acyclic directed mixed graph (ADMG) that represents this DAG marginal, using the latent projection algorithm [17]. This graph is shown in Fig. 1 (b); directed arrows in the resulting ADMG represent directed paths in the DAG where any intermediate nodes are unobserved (in this case there are no such paths, and all directed edges in the ADMG are directly inherited from the DAG). Similarly, bidirected arrows in the ADMG, such as $2 \leftrightarrow 5$, represent marginally d-connected paths in the DAG which start and end with arrowheads pointing away from the path, in this case $2 \leftarrow 6 \rightarrow 5$.

If we now use the nested Möbius parameters, described in more detail in subsequent sections, to parameterize the resulting ADMG, we will quickly discover that this results in a model of higher dimension relative to the dimension of DAG models which share their skeleton with this ADMG. For example, the binary nested Markov model of the graph in Fig. 1 (b) has 16 parameters, while both binary DAG models corresponding to graphs in Fig. 7 (a) and (b) have 11 parameters each.

This leads to a worry that a structure learning algorithm that tries to use nested Möbius parameters to recover an ADMG from data by means of a score method, such as BIC [13], which rewards fit and parameter parsimony, may prefer at low sample sizes incorrect independence models given by DAGs in preference to correct models given by ADMGs, simply because the DAG models compensate for their poor fit of the data with a much smaller parameter count. In fact, this precise issue has been observed in simulation studies reported in [15].

Addressing this problem with a Möbius parameterization is not easy, because Möbius parameters are strata or context-specific; in other words, the parameterization is not independent of how the states are labeled. For instance, some of the Möbius parameters representing confounding between X_2,X_4 and X_5 are: [1]

$$\theta_{\{2,4,5\}}(x_1, x_3) = p(0_4, 0_5 | x_3, 0_2, x_1)\, p(0_2 | x_1)$$

for all values of x_1, x_3. In a binary model, this gives 4 parameters. The kinds of regularities in the true generative process, which we may want to exploit to create a dimension reduction in our model, typically involve a lack of interactions among variables, or a latent confounder with a low dimensional state space. Such regularities may often not translate into constraints naturally expressible in terms of Möbius parameters.

To avoid this difficulty, we need to construct parameters for nested Markov models which represent various

[1]To save space, here and elsewhere we will write 1_i for an assignment of X_i to 1, and 0_i for an assignment to 0.

types of interactions among variables directly. In fact, parameters representing interactions are well known in log-linear models, of which undirected graphical models and certain regression models form a special case.

3 Log-linear Parameters for Undirected Models

We will use undirected graphical models, also known as Markov random fields, to illustrate log-linear models. A Markov random field over a multivariate binary state space $\mathfrak{X}_V$, is a set of densities $p(x_V)$ represented by an undirected graph $\mathcal{G}$ with vertices V, where

$$p(x_V) = \exp\left(\sum_{C \in \mathrm{cl}(\mathcal{G})} (-1)^{\|x_C\|_1} \lambda_C\right);$$

here $\mathrm{cl}(\mathcal{G})$ is the collection of (not necessarily maximal) cliques in the undirected graph, $\|\cdot\|_1$ is the L_1-norm, and λ_C is a *log-linear parameter*. Note that the parameter $\lambda_\emptyset$ ensures the expression is normalized.

Consider the undirected graph shown in Fig. 2. In this graph, all subsets of $\{1, 2, 3\}$, $\{2, 4\}$, and $\{4, 5, 6\}$ are cliques. The model represents densities where, conditional upon its adjacent nodes, each node is independent of all others. The log-linear parameter(s) corresponding to each such subset of size k can be viewed as representing k-way interactions among appropriate variables in the model. Setting some such interaction parameters to zero in a consistent way results in a model which still asserts the same conditional independences, but has a smaller parameter count, and with all strata in each clique treated symmetrically. For instance, if we were to set all parameters for cliques of size $k > 2$ to zero, so that there remained only parameters corresponding to vertices and individual edges, we would obtain a model known as a Boltzmann machine [1]. A similar idea had been used to give a sparse parameterization for discrete DAG models [12].

In the remainder of the paper, we describe nested Markov models, and give a log-linear parameterization for these models which contains similar parameters that may be set to zero. While in Markov random field models the parameters are associated with sets of nodes which form cliques in the corresponding undirected graph, in nested Markov models parameters will be associated with special sets of nodes in the corresponding ADMG called *intrinsic sets*. Further, log-linear parameterizations of this type can often incorporate individual-level continuous baseline covariates [5].

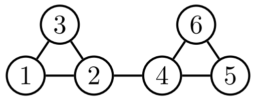


Figure 2: An undirected graph representing a log-linear model.

4 Graphs, Kernels, and Nested Markov Models

We now introduce the relevant background needed to define the nested Markov model.

A *directed mixed graph* $\mathcal{G}(V, E)$ is a graph with a set of vertices V and a set of edges E, where the edges may be directed ($\rightarrow$) or bidirected ($\leftrightarrow$). A directed cycle is a path of the form $x \rightarrow \cdots \rightarrow y$ along with an edge $y \rightarrow x$. An *acyclic* directed mixed graph (ADMG) is a mixed graph containing no directed cycles. An example is given in Fig. 1 (b).

Let a, b and d be vertices in a mixed graph $\mathcal{G}$. If $b \rightarrow a$ then we say that b is a *parent* of a, and a is a *child* of b. If $a \leftrightarrow b$ then a is said to be a *spouse* of b. A vertex a is said to be an *ancestor* of a vertex d if *either* there is a directed path $a \rightarrow \cdots \rightarrow d$ from a to d, *or* $a = d$; similarly d is said to be a *descendant* of a. The sets of parents, children, spouses, ancestors and descendants of a in $\mathcal{G}$ are written $\mathrm{pa}_\mathcal{G}(a)$, $\mathrm{ch}_\mathcal{G}(a)$, $\mathrm{sp}_\mathcal{G}(a)$, $\mathrm{an}_\mathcal{G}(a)$, $\mathrm{de}_\mathcal{G}(a)$ respectively. We apply these definitions disjunctively to sets, e.g. $\mathrm{an}_\mathcal{G}(A) = \bigcup_{a \in A} \mathrm{an}_\mathcal{G}(a)$.

4.1 Conditional ADMGs

A *conditional* acyclic directed mixed graph (CADMG) $\mathcal{G}(V, W, E)$ is an ADMG with a vertex set $V \cup W$, where $V \cap W = \emptyset$, subject to the restriction that for all $w \in W$, $\mathrm{pa}_\mathcal{G}(w) = \emptyset = \mathrm{sp}_\mathcal{G}(w)$. The vertices in V are the *random* vertices, and those in W are called *fixed*.

Whereas an ADMG with vertex set V represents a joint density $p(x_V)$, a conditional ADMG represents the Markov structure of a conditional density, or kernel $q_V(x_V|x_W)$. Following [8, p.46], we define a *kernel* to be a non-negative function $q_V(x_V|x_W)$ satisfying:

$$\sum_{x_V} q_V(x_V \mid x_W) = 1 \qquad \text{for all } x_W. \qquad (1)$$

We use the term 'kernel' and write $q_V(\cdot|\cdot)$ (rather than $p(\cdot|\cdot)$) to emphasize that these functions, though they satisfy (1) and thus most properties of conditional densities, are not in general formed via the usual operation of conditioning on the event $X_W = x_W$. To conform with standard notation for densities, for every $A \subseteq V$

let

$$q_V(x_A|x_W) \equiv \sum_{V\setminus A} q_V(x_V|x_W),$$

$$q_V(x_{V\setminus A}|x_{W\cup A}) \equiv \frac{q_V(x_V|x_W)}{q_V(x_A|x_W)}.$$

For a CADMG $\mathcal{G}(V, W, E)$ we consider collections of random variables $(X_v)_{v\in V}$ indexed by variables $(X_w)_{w\in W}$; throughout this paper the random variables take values in finite discrete sets $(\mathfrak{X}_v)_{v\in V}$ and $(\mathfrak{X}_w)_{w\in W}$. For $A \subseteq V \cup W$ we let $\mathfrak{X}_A \equiv \times_{u\in A}(\mathfrak{X}_u)$, and $X_A \equiv (X_v)_{v\in A}$. That we will always hold the variables in W fixed is precisely why we do not permit edges between vertices in W.

An ADMG $\mathcal{G}(V, E)$ may be seen as a CADMG in which $W = \emptyset$. In this manner, though we will state subsequent definitions for CADMGs, they also apply to ADMGs.

The *induced subgraph* of a CADMG $\mathcal{G}(V, W, E)$ given by a set A, denoted $\mathcal{G}_A$, consists of $\mathcal{G}(V\cap A, W\cap A, E_A)$ where E_A is the set of edges in $\mathcal{G}$ with both endpoints in A. In forming $\mathcal{G}_A$, the status of the vertices in A with regard to whether they are in V or W is preserved.

4.2 Districts and Markov Blankets

A set C is *connected* in $\mathcal{G}$ if every pair of vertices in C is joined by a path such that every vertex on the path is in C. For a given CADMG $\mathcal{G}(V, W, E)$, denote by $(\mathcal{G})_\leftrightarrow$ the CADMG formed by removing all directed edges from $\mathcal{G}$. A set connected in $(\mathcal{G})_\leftrightarrow$ is called *bidirected connected*.

For a vertex $x \in V$, the *district* (or c-component) of x, denoted by $\text{dis}_\mathcal{G}(x)$, is the maximal bidirected connected set containing x. For instance in the ADMG shown in Fig. 1 (b), the district of node 2 is $\{2, 4, 5\}$. Districts in a CADMG form a partition of V; vertices in W are excluded by definition. In a DAG $\mathcal{G}(V, E)$ the set of districts is the set of all single element sets $\{v\} \subseteq V$.

A set of vertices A in $\mathcal{G}$ is called *ancestral* if $a \in A \Rightarrow \text{an}_\mathcal{G}(a) \subseteq A$. In a CADMG $\mathcal{G}(V, W, E)$, if A is an ancestral subset of $V \cup W$ in $\mathcal{G}$, $t \in A\cap V$, and $\text{ch}_\mathcal{G}(t) \cap A = \emptyset$, then the *Markov blanket of t in A* is defined as:

$$\text{mb}_\mathcal{G}(t, A) \equiv \text{pa}_\mathcal{G}\Big(\text{dis}_{\mathcal{G}_A}(t)\Big) \cup \Big(\text{dis}_{\mathcal{G}_A}(t) \setminus \{t\}\Big).$$

4.3 The fixing operation and fixable vertices

We now introduce a 'fixing' operation on a CADMG which has the effect of transforming a random vertex into a fixed vertex, thereby changing the graph. However, this operation is only applicable to a subset of the vertices, which we term the (potentially) fixable vertices.

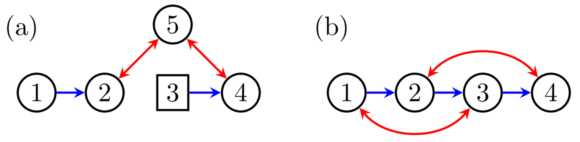


Figure 3: (a) The graph from Fig. 1 (b) after fixing 3. (b) An ADMG inducing a non-trivial nested Markov model.

Definition 1 *Given a CADMG $\mathcal{G}(V, W, E)$ the set of* fixable vertices *is*

$$\mathbb{F}(\mathcal{G}) \equiv \{v \mid v \in V, \text{dis}_\mathcal{G}(v) \cap \text{de}_\mathcal{G}(v) = \{v\}\}.$$

In words, v is fixable in $\mathcal{G}$ if there is no vertex v^* that is both a descendant of v and in the same district as v. For the graph in Fig. 1 (b), the vertex 2 is not fixable, because 4 is both its descendant and in the same district; all the other vertices are fixable.

Definition 2 *Given a CADMG $\mathcal{G}(V, W, E)$, and a kernel $q_V(X_V \mid X_W)$, with every $r \in \mathbb{F}(\mathcal{G})$ we associate a* fixing transformation *ϕ_r on the pair $(\mathcal{G}, q_V(X_V \mid X_W))$ defined as follows:*

$$\phi_r(\mathcal{G}) \equiv \mathcal{G}^*(V \setminus \{r\}, W \cup \{r\}, E_r),$$

where E_r is the subset of edges in E that do not have arrowheads into r, and

$$\phi_r(q_V(x_V \mid x_W); \mathcal{G}) \equiv \frac{q_V(x_V \mid x_W)}{q_V(x_r \mid x_{\text{mb}_\mathcal{G}(r, \text{an}_\mathcal{G}(\text{dis}_\mathcal{G}(r)))})}$$

Returning to the ADMG in Fig. 1 (b), fixing 3 in the graph means removing the edge $2 \to 3$, while fixing x_3 in $p(x_1, x_2, x_3, x_4, x_5)$ means dividing this marginal density by $q_{1,2,3,4,5}(x_3 \mid x_2) = p(x_3|x_2)$. The resulting CADMG, shown in Fig. 3 (a), represents the resulting kernel $q_{1,2,4,5}(x_1, x_2, x_4, x_5|x_3)$.

We use $\circ$ to indicate composition of operations in the natural way, so that: $\phi_r \circ \phi_s(\mathcal{G}) \equiv \phi_r(\phi_s(\mathcal{G}))$ and

$$\begin{aligned}&\phi_r \circ \phi_s(q_V(X_V|X_W); \mathcal{G})\\ &\qquad \equiv \phi_r\left(\phi_s\left(q_V(X_V|X_W); \mathcal{G}\right); \phi_s(\mathcal{G})\right).\end{aligned}$$

4.4 Reachable and Intrinsic Sets

In order to define the nested Markov model, we will need to define special classes of vertex sets in ADMGs.

Definition 3 *A CADMG $\mathcal{G}(V, W)$ is* reachable *from an ADMG $\mathcal{G}^*(V \cup W)$ if there is an ordering of the vertices in $W = \langle w_1, \ldots, w_k \rangle$, such that for $j = 1, \ldots, k$,*

$$w_1 \in \mathbb{F}(\mathcal{G}^*) \text{ and for } j = 2, \ldots, k,$$
$$w_j \in \mathbb{F}(\phi_{w_{j-1}} \circ \cdots \circ \phi_{w_1}(\mathcal{G}^*)).$$

A subgraph is reachable if, under some ordering, each vertex w_i is fixable in $\phi_{w_{i-1}}(\cdots \phi_{w_2}(\phi_{w_1}(\mathcal{G}^*)) \cdots)$.

Fixing operations do not in general commute, and thus only some orderings are valid for fixing a particular set. For example, in the ADMG shown in Fig. 1 (b), the set $\{2, 3\}$ may be fixed, but only under the ordering where 3 is fixed first, to yield the CADMG shown in Fig. 3 (a), and then 2 is fixed in this CADMG. Fixing 2 first in Fig. 1 (b) is not valid, because 4 is both a descendant of 2 and in the same district as 2 in that graph, and thus 2 is not fixable.

If a CADMG $\mathcal{G}(V, W)$ is reachable from $\mathcal{G}^*(V \cup W)$, we say that the set V is reachable in $\mathcal{G}^*$. A reachable set may be obtained by fixing vertices using more than one valid sequence. We will denote any valid composition of fixing operations that fixes a set A by ϕ_A if applied to the graph, and by ϕ_{X_A} if applied to a kernel. With a slight abuse of notation (though justified as we will later see) we suppress the precise fixing sequence chosen.

Definition 4 *A set of vertices S is* intrinsic *in $\mathcal{G}$ if it is a district in a reachable subgraph of $\mathcal{G}$. The set of intrinsic sets in an ADMG $\mathcal{G}$ is denoted by $\mathcal{I}(\mathcal{G})$.*

For example, in the graph in Fig. 1 (b), the set $\{2, 4, 5\}$ is intrinsic (and reachable), while the set $\{1, 2, 4, 5\}$ is reachable but not intrinsic.

In any DAG $\mathcal{G}(V, E)$, $\mathcal{I}(\mathcal{G}) = \{\{x\} | x \in V\}$, while in any bidirected graph $\mathcal{G}$, $\mathcal{I}(\mathcal{G})$ is equal to the set of all connected sets in $\mathcal{G}$.

4.5 Nested Markov Models

Just as for DAG models, nested Markov models may be defined via one of several equivalent Markov properties. These properties are all *nested* in the sense that they apply recursively to either reachable or intrinsic sets derived from an ADMG. In particular, there is a nested analogue of the global Markov property for DAGs (d-separation), the local Markov property for DAGs (which asserts that variables are independent of non-descendants given parents), and the moralization-based property for DAGs. These definitions appear and are proven equivalent in [11]. It is possible to associate a unique ADMG with a particular marginal DAG model, and a nested Markov model associated with this ADMG will recover all independences which hold in the marginal DAG [11].

We now define a nested factorization on probability distributions represented by ADMGs using special sets of nodes called 'recursive heads' and 'tails.'

Definition 5 *For an intrinsic set $S \in \mathcal{I}(\mathcal{G})$ of a CADMG $\mathcal{G}$, define the recursive head (rh) as:* $\text{rh}(S) \equiv \{x \in S \mid \text{ch}_{\mathcal{G}}(x) \cap S = \emptyset\}$.

Definition 6 *The* tail *associated with a recursive head H of an intrinsic set S in a CADMG $\mathcal{G}$ is given by:* $\text{tail}(H) \equiv (S \setminus H) \cup \text{pa}_{\mathcal{G}}(S)$.

In the graph in Fig. 1 (b), the recursive head of the intrinsic set $\{2, 4, 5\}$ is equal to the set itself, while the tail is $\{1, 3\}$.

A kernel $q_V(X_V | X_W)$ satisfies the *head factorization property* for a CADMG $\mathcal{G}(V, W, E)$ if there exist kernels $\{f_S(X_H | X_{\text{tail}(H)}) \mid S \in \mathcal{I}(\mathcal{G}), H = \text{rh}_{\mathcal{G}}(S)\}$ such that

$$q_V(X_V | X_W) = \prod_{\substack{H \in [\![V]\!]_{\mathcal{G}} \\ S : \text{rh}_{\mathcal{G}}(S) = H}} f_S(X_H | X_{\text{tail}(H)}) \qquad (2)$$

where $[\![V]\!]_{\mathcal{G}}$ is a partition of V into heads given in [14].

Let $\mathbb{G}(\mathcal{G}) \equiv \{(\mathcal{G}^*, \mathbf{w}^*) \mid \mathcal{G}^* = \phi_{\mathbf{w}^*}(\mathcal{G})\}$ for an ADMG $\mathcal{G}$. That is, $\mathbb{G}(\mathcal{G})$ is the set of valid fixing sequences and the CADMGs that they reach. The same graph may be reached by more than one sequence $\mathbf{w}^*$. We say that a distribution $p(x_V)$ obeys the *nested head factorization property* for $\mathcal{G}$ if for all $(\mathcal{G}^*, \mathbf{w}^*) \in \mathbb{G}(\mathcal{G})$, the kernel $\phi_{\mathbf{w}^*}(p(X_V); \mathcal{G})$ obeys the head factorization for $\phi_{\mathbf{w}^*}(\mathcal{G}) \equiv \mathcal{G}^*$. We denote the set of such distributions by $\mathcal{P}_h^n(\mathcal{G})$. Nested Markov models have been defined via a nested district factorization criterion [15], and a number of Markov properties [11]. The head factorization is another way of defining the nested Markov model due to the following result.

Theorem 7 *The set $\mathcal{P}_h^n(\mathcal{G})$ is the nested Markov model of $\mathcal{G}$.*

Our decision to suppress the precise fixing sequence from the fixing operation applied to sets is justified, due to the following result.

Theorem 8 *If $p(x_V)$ is in the nested Markov model of $\mathcal{G}$, then for any reachable set A in $\mathcal{G}$, any valid fixing sequence on $V \setminus A$ gives the same CADMG over A, and the same kernel $q_A(x_A | x_{V \setminus A})$ obtained from $p(x_V)$.*

4.6 A Möbius Parameterization of Binary Nested Markov Models

We now give the original parameterization for binary nested Markov models. The approach generalizes in a straightforward way to finite discrete state spaces. Multivariate binary distributions in the nested Markov model for an ADMG $\mathcal{G}$ may be parameterized by the following:

Definition 9 *The* nested Möbius parameters *associated with a CADMG $\mathcal{G}$ are a set of functions: $\mathfrak{Q}_{\mathcal{G}} \equiv \big\{q_S(X_H = \mathbf{0} \,|\, x_{\mathrm{tail}(H)})$ for $H = \mathrm{rh}(S), S \in \mathcal{I}(\mathcal{G})\big\}$.*

Intuitively, a parameter $q_S(X_H = \mathbf{0}|x_{\mathrm{tail}(H)})$ is the probability that the variable set X_H assumes values $\mathbf{0}$ in a kernel obtained from $p(x_V)$ by fixing $X_{V\setminus S}$, and conditioning on $X_{\mathrm{tail}(H)}$. As a shorthand, we will denote the parameter $q_S(X_H = \mathbf{0}|x_{\mathrm{tail}(H)})$ by $\theta_H(x_{\mathrm{tail}(H)})$.

Definition 10 *Let $\nu : V \cup W \mapsto \{0,1\}$ be an assignment of values to the variables indexed by $V \cup W$. Define $\nu(T)$ to be the values assigned to variables indexed by a subset $T \subseteq V \cup W$. Let $\nu^{-1}(0) = \{v \mid v \in V, \nu(v) = 0\}$.*

A distribution $P(X_V \mid X_W)$ is said to be parameterized by *the set $\mathfrak{Q}_{\mathcal{G}}$, for CADMG $\mathcal{G}$ if:*

$$p(X_V = \nu(V) \,|X_W = \nu(W)) = \sum_{B\,:\,\nu^{-1}(0)\cap V \subseteq B \subseteq V} (-1)^{|B\setminus \nu^{-1}(0)|} \times \prod_{H \in [\![B]\!]_{\mathcal{G}}} \theta_H(X_{\mathrm{tail}(H)} = \nu(\mathrm{tail}(H))) \quad (3)$$

where the empty product is defined to be 1.

For example, the graph shown in Fig. 3 (b) representing a model over binary random variables X_1, X_2, X_3, X_4 is parameterized by the following sets of parameters:

$$\begin{aligned}
\theta_1 &= p(0_1)\\
\theta_2(x_1) &= p(0_2|x_1)\\
\theta_{1,3}(x_2) &= p(0_3|x_2, 0_1)p(0_1)\\
\theta_3(x_2) &= \sum_{x_1} p(0_3|x_2, x_1)p(x_1)\\
\theta_{2,4}(x_1, x_3) &= p(0_4|x_3, 0_2, x_1)p(0_2|x_1)\\
\theta_4(x_3) &= \sum_{x_2} p(0_4|x_3, x_2, x_1)p(x_2|x_1).
\end{aligned}$$

The total number of parameters is $1+2+2+2+4+2 = 13$, which is 2 fewer than a saturated parameterization of a 4 node binary model, which contains $2^4 - 1 = 15$ parameters. The two missing parameters reflect the fact that $\theta_4(x_3)$ does not depend on x_1, which is a constraint induced by the absence of the edge from 1 to 4 in Fig. 3 (b). Note that this constraint is not a conditional independence. In fact, no conditional independences over variables corresponding to vertices $1, 2, 3, 4$ are advertised in Fig. 3 (b).

This parameterization maps θ_H parameters to probabilities in a CADMG via the inverse Möbius transform given by (3), and generalizes both the standard Markov parameterization of DAGs in terms of parameters of the form $p(x_i = 0 \,|\, \mathrm{pa}(x_i))$, and the parameterization of bidirected graph models given in [3].

5 A Log-linear Parameterization of Nested Markov Models

We begin by defining a set of objects which are functions of the observed density, and which will serve as our parameters.

Definition 11 *Let $\mathcal{G}(V, E)$ be an ADMG and $p(x_V)$ a density over a set of binary random variables X_V in the nested Markov model of $\mathcal{G}$. For any $S \in \mathcal{I}(\mathcal{G})$, let $M = S \cup \mathrm{pa}_{\mathcal{G}}(S)$, $A \subseteq M$ (with $A \cap S \neq \emptyset$), and let $q_S(x_S|x_{M\setminus S}) = \phi_{V\setminus S}(p(x_V); \mathcal{G})$ be the associated kernel. Then define*

$$\lambda_A^M = \frac{1}{2^{|M|}} \sum_{x_M} (-1)^{\|x_A\|_1} \log q_S(x_S|x_{M\setminus S}),$$

to be the nested log-linear parameter associated with A in S. *Further let $\Lambda(\mathcal{G})$ be the collection*

$$\{\lambda_A^M \,|\, S \in \mathcal{I}(\mathcal{G}), M = S \cup \mathrm{pa}_{\mathcal{G}}(S), \mathrm{rh}_{\mathcal{G}}(S) \subseteq A \subseteq M\}$$

of these log-linear parameters. We call $\Lambda(\mathcal{G})$ the nested ingenuous parameterization *of $\mathcal{G}$.*

This parameterization is based on the graphical concepts of recursive heads and corresponding tails. We call the parameterization 'nested ingenuous' due to its similarity to a marginal log-linear parameterization called ingenuous in [6], and in contrast to other log-linear parameterizations which may exist for nested Markov models. Marginal model parameterizations of this type were first introduced in [2]. This definition extends easily to non-binary discrete data, in which case some parameters λ_A^M become collections of parameters.

As an example, consider the graph shown in Fig. 3 (b) which represents a binary nested Markov model. The nested ingenuous parameters associated with the marginal $p(x_1)$ and conditional $p(x_2|x_1)$ are

$$\begin{aligned}
\lambda_1^1 &= \frac{1}{2} \log \frac{p(0_1)}{p(1_1)}\\
\lambda_2^{21} &= \frac{1}{4} \log \frac{p(0_2|0_1) \cdot p(0_2|1_1)}{p(1_2|0_1) \cdot p(1_2|1_1)}\\
\lambda_{21}^{21} &= \frac{1}{4} \log \frac{p(0_2|0_1) \cdot p(1_2|1_1)}{p(1_2|0_1) \cdot p(0_2|1_1)}
\end{aligned}$$

whereas parameters associated with the kernel $q_4(x_4|x_3) = \sum_{x_2} p(x_4|x_3, x_2, x_1)p(x_2|x_1)$ are

$$\lambda_4^{43} = \frac{1}{4}\log\frac{q_4(0_4|0_3)\cdot q_4(0_4|1_3)}{q_4(1_4|0_3)\cdot q_4(1_4|1_3)}$$
$$\lambda_{43}^{43} = \frac{1}{4}\log\frac{q_4(0_4|0_3)\cdot q_4(1_4|1_3)}{q_4(1_4|0_3)\cdot q_4(0_4|1_3)}$$

A parameter λ_A^M, where M is the union of a head H and its tail T, can be viewed, by analogy with similar clique parameters in undirected log-linear models, as a $|A|$-way interaction between the vertices in A, within the kernel corresponding to M. For instance the kernel $q_{2,4}(x_2, x_4|x_1, x_3) = p(x_4|x_3, x_2, x_1)\,p(x_2|x_1)$,[2] makes an appearance in 4 parameters in a binary model: λ_{24}^{1234}, λ_{124}^{1234}, λ_{234}^{1234}, and λ_{1234}^{1234}. If we set λ_{1234}^{1234} to 0, we claim there is no 4-way interaction between X_1, X_2, X_3, X_4 in the kernel.

It can be shown that while the Möbius parameterization of the graph in Fig. 3 (b) is variation dependent, the nested ingenuous parameterization of the same graph is variation independent. This is not true in general. In particular both parameterizations for the graph in Fig. 1 (b) are variation dependent.

6 Main Results

In this section we prove that the nested ingenuous parameters indeed parameterize discrete nested Markov models. We start with an intermediate result.

Lemma 12 *Let $H \subseteq M$ and $q(x_H \,|\, x_{M\setminus H})$ be a kernel. Then q is smoothly parameterized by the collection of NLL parameters $\{\lambda_A^M \,|\, H \subseteq A \subseteq M\}$ together with the $(|H|-1)$-dimensional margins of q, $q(x_{H\setminus\{v\}} \,|\, x_{M\setminus H}), v \in H$.*

Proof: The proof is essentially identical to the proof of Lemma 4.4 in [6]. □

6.1 The Main Result

We now define a partial order on heads and use this order to inductively establish the main result.

Definition 13 *Let $\prec_{\mathcal{I}(\mathcal{G})}$ be the partial order on heads, H_i, of intrinsic sets, S_i, in $\mathcal{G}$ such that $H_i \prec_{\mathcal{I}(\mathcal{G})} H_j$ whenever $S_i \subset S_j$.*

Theorem 14 *The nested ingenuous parameterization of an ADMG $\mathcal{G}$ with nodes V parameterizes precisely those distributions $p(x_V)$ obeying the nested global Markov property with respect to $\mathcal{G}$.*

[2]This kernel, viewed causally, is $p(x_2, x_4|\text{do}(x_1, x_3))$.

Proof: Let $\prec_{\mathcal{I}(\mathcal{G})}$ be the partial ordering on heads given in Definition 13. We proceed by induction on this ordering. For the base case, we know that singleton heads $\{h\}$ with empty tails are parameterized by λ_h^h. If a singleton head has a non-empty tail, the conclusion follows immediately by Lemma 12.

Now, suppose that we wish to find the kernel with a non-singleton head $H^\dagger$ and a tail $T^\dagger$ corresponding to the intrinsic set $S^\dagger$. Assume, by inductive hypothesis, that we have already obtained the kernels with all heads H such that $H \prec_{\mathcal{I}(\mathcal{G})} H^\dagger$. We claim this is sufficient to give the $(|H^\dagger|-1)$-dimensional marginal kernels $q_{S^\dagger}(x_{H^\dagger\setminus\{v\}}|x_{T^\dagger})$, for all $v \in H^\dagger$.

Fix a particular $v \in H^\dagger$. The set $S^\dagger \setminus \{v\}$ is reachable, since $V \setminus S^\dagger$ is a set with a valid fixing sequence, and any $v \in H^\dagger$ has no children in $S^\dagger$ in $\phi_{V\setminus S^\dagger}(\mathcal{G})$ so is fixable in $\phi_{V\setminus S^\dagger}(\mathcal{G})$. Theorem 7 and Theorem 8 imply that for every reachable set A, (2) holds. Hence:

$$q_{S^\dagger}(x_{S^\dagger\setminus\{v\}}|x_{V\setminus(S^\dagger\setminus\{v\})}) = \prod_{\substack{H\in[\![S^\dagger\setminus\{v\}]\!]_{\mathcal{G}}\\ S:\text{rh}_{\mathcal{G}}(S)=H}} q_S(x_H|x_{\text{tail}(H)}). \tag{4}$$

For any S such that $\text{rh}_{\mathcal{G}}(S) = H$, and $H \subseteq S^\dagger \setminus \{v\}$, $H \prec_{\mathcal{I}(\mathcal{G})} H^\dagger$, hence by the induction hypothesis, the kernel $q_S(x_S|x_{pa(S)\setminus S})$ is already obtained, and all kernels which appear in (4) can be derived by conditioning from some such $q_S(x_S|x_{pa(S)\setminus S})$. The desired kernel $q_{S^\dagger}(x_{H^\dagger\setminus\{v\}}|x_{T^\dagger})$ can itself be obtained from $q_{S^\dagger}(x_{S^\dagger\setminus\{v\}}|x_{\text{pa}(S^\dagger)\setminus S^\dagger})$ by conditioning.

We can repeat this argument for any $v \in H^\dagger$. Finally, the nested ingenuous parameterization contains $\lambda_A^{H^\dagger\cup T^\dagger}$ for $H^\dagger \subseteq A \subseteq H^\dagger\cup T^\dagger$. The result then follows by Lemma 12. □

7 Simulations

To illustrate the utility of setting higher order parameters to zero ('removing'), we present a simulation study based on the ADMG in Fig. 5 (b). This graph is a special case of two bidirected chains of k vertices each, with a path of directed edges alternating between the chains, for $k = 4$. The number of parameters in the relevant binary nested Markov model grows exponentially with k in graphs of this type.

Consider also the latent variable model defined by replacing each bidirected edge with an independent latent variable shown in Fig. 5 (a), so that $1 \leftrightarrow 3$ becomes $1 \leftarrow 9 \rightarrow 3$. If the state space of each latent variable is the same and fixed, then the number of parameters in this hidden variable DAG model grows only linearly in k. This suggests that the nested Markov model may include higher order parameters which are

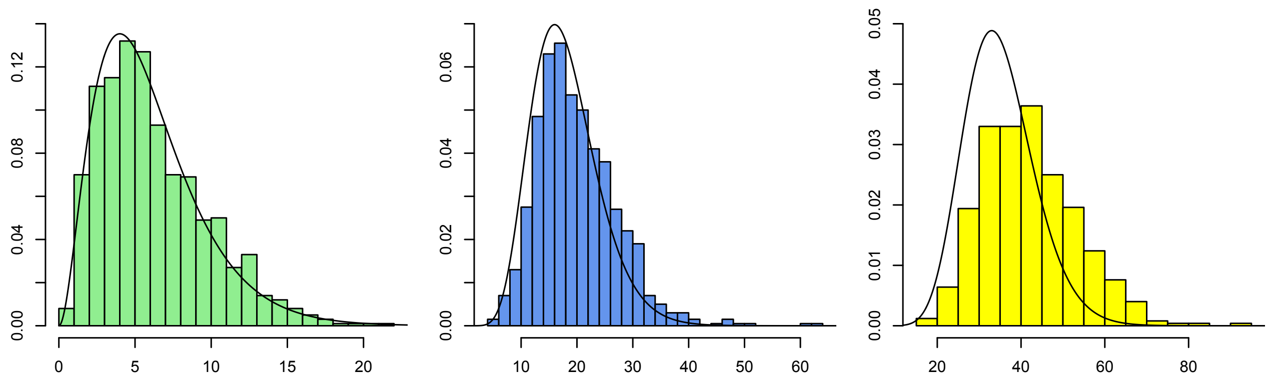


Figure 4: Histograms showing the increase in deviance associated with setting to zero any nested log-linear parameters with effects higher than orders (from left to right) seven, six and five respectively. This corresponds to removing 6, 18 and 35 parameters respectively; the relevant χ^2 density is plotted in each case.

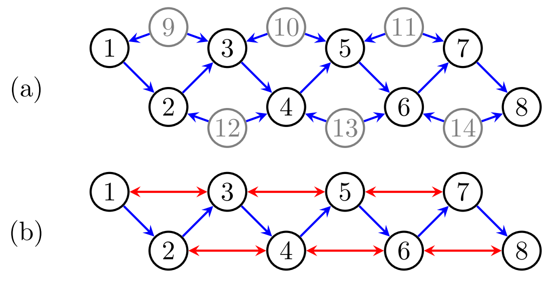


Figure 5: (a) A hidden variable DAG used to generate samples for Section 7. (b) The latent projection of this generating DAG.

not really necessary in this case (though the higher order parameters may become necessary again if the state space of latent variables grows).

We generated distributions from the latent variable model associated with the DAG in Fig. 5 (a) as follows: each of the six latent variables takes one of three states with equal probability, and each observed variable takes the value 0 with a probability generated as an independent uniform random variable on $(0, 1)$, conditional upon each possible value of its parents.

For each of 1,000 distributions produced independently using this method, we generated a dataset of size 5,000. We then fitted the nested model generated by the graph in Fig. 5 (b) to each dataset by maximum likelihood estimation, using a variation of an algorithm found in [5], and measured the increase in deviance associated with zeroing any nested ingenuous parameters corresponding to effects above a certain order. If these parameters were truly zero, we would expect the increase to follow a χ^2-distribution with an appropriate number of degrees of freedom; the first two histograms in Fig. 4 demonstrate that the distribution of actual increases in deviance looks much like the relevant χ^2-distribution if we remove interactions of order 6 and higher. The third histogram shows that this starts to break down slightly when 5-way interactions are also zeroed.

These results suggest that higher order parameters are often not useful for explaining finite datasets, and more parsimonious models can be obtained by removing them; a similar simulation was performed for the Markov case in [6].

7.1 Distinguishing Graphs

The use of score-based search methods for recovering nested Markov models had been investigated [15]. It was found that relatively large sample sizes were required to reliably recover the correct graph, even in examples with only 4 or 5 binary nodes and after ensuring that the underlying distributions were approximately faithful to the true graph. One phenomenon identified was that incorrect but more parsimonious graphs, especially DAGs, tended to have lower BIC scores than the correct models, which include higher order parameters. Although BIC is guaranteed to be smaller on the correct model asymptotically, in finite samples it applies strong penalties for having additional parameters with little explanatory power.

Here we present a simulation to show how the new parameterization can help to overcome this difficulty. Using the method described in the previous subsection, we generated 1,000 multivariate binary distributions which were nested Markov with respect to the graph in Fig. 1 (b). For each distribution we generated a dataset, and fitted the data to the correct model, which has 16 parameters, as well as the two DAGs given in Fig. 7 (a) and (b), which each have

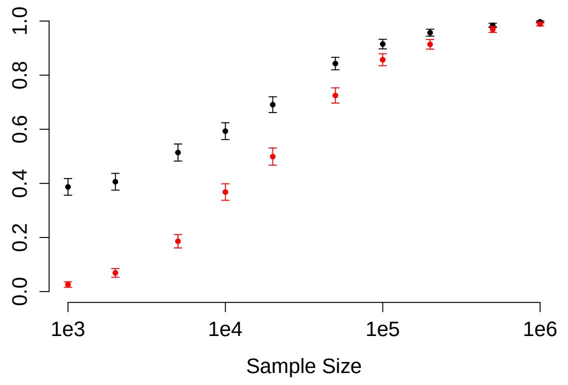


Figure 6: From the experiment in Section 7.1: in red, the proportion of times graph in Fig. 1 (b) had lower BIC than the DAGs in Fig. 7, for varying sample sizes; in black, the proportion of times some restricted version of this model had a lower BIC than any restricted versions of either DAG.

11 parameters. This was repeated at various sample sizes.

The plot in Fig. 6 shows, in red, the proportion of times in which the BIC score for the correct model was lower than that for each of the DAGs, at various sample sizes. The correct graph only has the lowest BIC score of the three graphs on less than 3% of runs at sample size of $n = 1{,}000$, increasing to around 50% for $n = 20{,}000$.

In addition to the full models, we fitted the datasets to versions of the models with higher order parameters removed; the graph in Fig. 1 (b) can be restricted by zeroing the 5-way parameter (leaving 15 free parameters), the 4-way and and above (13 params), or 3-way and above (10 params). Similarly we can restrict the DAGs to have no 3-way effects, giving each model 10 free parameters. Fig. 6 shows, in black, the proportion of times that one of these restricted versions of the true model had a lower BIC than any version of either DAG model. We see that the correct graph has the lowest score in 40% of runs at $n = 1{,}000$, rising to around 70% at $n = 20{,}000$. Note that these results should not be compared directly to those in [15], since the single ground truth law used in that paper was generated so as to ensure faithfulness to the correct graph, whereas we are randomly sampling multiple laws without bothering to ensure any particular properties in these laws other than consistency with the underlying DAG.

These results suggest that these submodels of the nested model may be advantageous in recovering the correct graphical structure using score-based methods.

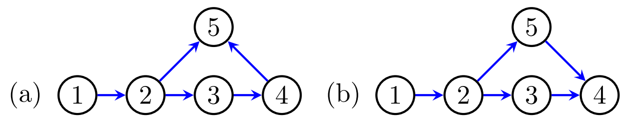


Figure 7: (a) and (b) two DAGs with the same skeleton as the graph in Fig. 1 (b).

Note that determining which higher order parameters should be set to zero for a given data set and sample size remains non-trivial. Automatic selection might be possible with an L_1-penalized approach [16, 4].

8 Discussion and Conclusions

We have introduced a new log-linear parameterization of nested Markov models over discrete state spaces. The log-linear parameters correspond to 'interactions' in kernels obtained after an iterative application of truncation and marginalization steps (informally 'interactions in interventional densities'). By contrast the Möbius parameters [15] correspond to context specific effects in kernels (informally 'context specific causal effects').

We have shown by means of a simulation study that in cases where data is generated from a marginal of a DAG with 'weak confounders', we can reduce the dimension of the model by ignoring higher order interaction parameters, while retaining the advantages of nested Markov models compared to modeling weak confounding directly in a DAG.

Though there is no efficient, closed form mapping from ingenuous parameters to either Möbius parameters or standard probabilities, this is a smaller disadvantage than it may seem. This is because in cases where the ingenuous parameterization was used to select a particular submodel based on a dataset, we may still reparameterize and use Möbius parameters, or even standard joint probabilities if desired. Moreover, this reparameterization step need only be performed once, compared to multiple calls to a fitting procedure which identified the particular graph corresponding to our submodel in the first place.

The ingenuous and the Möbius parameterizations are thus complementary. The natural application of the ingenuous parameterization is in learning graph structure from data in situations where many samples are not available, but we expect most confounding to be weak. The natural application of the Möbius parameterization is the support of probabilistic and causal inference in a particular graph [14, 15], in cases where an efficient mapping from parameters to joint probabilities is important.

References

[1] D. H. Ackley, G. E. Hinton, and T. J. Seinowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985.

[2] W. P. Bergsma and T. Rudas. Marginal models for categorical data. *Annals of Statistics*, 30(1):140–159, 2002.

[3] M. Drton and T. S. Richardson. Binary models for marginal independence. *Journal of the Royal Statistical Society (Series B)*, 70(2):287–309, 2008.

[4] R. J. Evans. *Parametrizations of Discrete Graphical Models*. PhD thesis, Department of Statistics, University of Washington, 2011.

[5] R. J. Evans and A. Forcina. Two algorithms for fitting constrained marginal models. *Computational Statistics & Data Analysis*, 66:1–7, 2013.

[6] R. J. Evans and T. S. Richardson. Marginal log-linear parameters for graphical Markov models. *Journal of the Royal Statistical Society (Series B) (to appear)*, 2013.

[7] D. Geiger, D. Heckerman, H. King, and C. Meek. Stratified exponential families: Graphical models and model selection. *Annals of Statistics*, 29(2):505–529, 2001.

[8] S. L. Lauritzen. *Graphical Models*. Oxford, U.K.: Clarendon, 1996.

[9] N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3):1436–1462, 2006.

[10] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan and Kaufmann, San Mateo, 1988.

[11] T. S. Richardson, J. M. Robins, and I. Shpitser. Nested Markov properties for acyclic directed mixed graphs. In *28th Conference on Uncertainty in Artificial Intelligence (UAI-12)*. AUAI Press, 2012.

[12] T. Rudas, W. P. Bergsma, and R. Nemeth. Parameterization and estimation of path models for categorical data. In *Proceedings in Computational Statistics, 17th Symposium*, pages 383–394. Physica-Verlag HD, 2006.

[13] G. E. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

[14] I. Shpitser, T. S. Richardson, and J. M. Robins. An efficient algorithm for computing interventional distributions in latent variable causal models. In *27th Conference on Uncertainty in Artificial Intelligence (UAI-11)*. AUAI Press, 2011.

[15] I. Shpitser, T. S. Richardson, J. M. Robins, and R. J. Evans. Parameter and structure learning in nested Markov models. In *UAI Workshop on Causal Structure Learning 2012*, 2012.

[16] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58(1):267–288, 1996.

[17] T. S. Verma and Judea Pearl. Equivalence and synthesis of causal models. Technical Report R-150, Department of Computer Science, University of California, Los Angeles, 1990.

Scalable Matrix-valued Kernel Learning for High-dimensional Nonlinear Multivariate Regression and Granger Causality

Vikas Sindhwani
IBM Research
Yorktown Heights, NY 10598
`vsindhw@us.ibm.com`

Hà Quang Minh
Istituto Italiano di Tecnologia
Genova, 16163, Italy
`minh.haquang@iit.it`

Aurélie C. Lozano
IBM Research
Yorktown Heights, NY 10598
`aclozano@us.ibm.com`

Abstract

We propose a general matrix-valued multiple kernel learning framework for high-dimensional nonlinear multivariate regression problems. This framework allows a broad class of mixed norm regularizers, including those that induce sparsity, to be imposed on a dictionary of vector-valued Reproducing Kernel Hilbert Spaces. We develop a highly scalable and eigendecomposition-free algorithm that orchestrates two inexact solvers for simultaneously learning both the input and output components of separable matrix-valued kernels. As a key application enabled by our framework, we show how high-dimensional causal inference tasks can be naturally cast as sparse function estimation problems, leading to novel nonlinear extensions of a class of Graphical Granger Causality techniques. Our algorithmic developments and extensive empirical studies are complemented by theoretical analyses in terms of Rademacher generalization bounds.

1 Introduction

Consider the problem of estimating an unknown nonlinear function, $f : \mathcal{X} \mapsto \mathcal{Y}$, from labeled examples, where $\mathcal{Y}$ is a "structured" output space [6]. In principle, $\mathcal{Y}$ may be endowed with a general Hilbert space structure, though we focus on the multivariate regression setting, where $\mathcal{Y} \subseteq \mathbb{R}^n$. Such problems can be naturally formulated as Tikhonov Regularization [35] in a suitable *vector-valued* Reproducing Kernel Hilbert Space (RKHS) [25, 1]. The theory and formalism of vector-valued RKHS can be traced as far back as the work of Laurent Schwarz in 1964 [32]. Yet, vector-valued extensions of kernel methods have not found widespread application, in stark contrast to the versatile popularity of their scalar cousins. We believe that two key factors are responsible:

- The kernel function is much more complicated - it is *matrix-valued* in this setting. Its choice turns into a daunting model selection problem. Contrast this with the scalar case where Gaussian or Polynomial kernels are a default choice requiring only a few hyperparameters to be tuned.
- The associated optimization problems, in the most general case, have much greater computational complexity than in the scalar case. For example, a vector-valued Regularized Least Squares (RLS) solver would require cubic time in the number of samples *multiplied by the number of output coordinates.*

Scalable kernel learning therefore becomes a basic necessity – an unavoidable pre-requisite – for even considering vector-valued RKHS methods for an application at hand. Our contributions in this paper are as follows:

- We propose a general framework for function estimation over a dictionary of vector-valued RKHSs where a broad family of variationally defined regularizers, including sparsity inducing norms, serve to optimally combine a collection of matrix-valued kernels. As such our framework may be viewed as providing generalizations of scalar multiple kernel learning [20, 26, 21, 29] and associated structured sparsity algorithms [8].
- We specialize our framework to the class of *separable kernels* [1] which are of interest due to their universality [9], conceptual simplicity and potential for scalability. Separable matrix-valued kernels are composed of a *scalar input kernel* component and a *positive semi-definite output matrix* component (to be formally defined later). We provide a full resolution of the kernel learning problem in this setting by jointly estimating both components together. This is in contrast to recent efforts [12, 18] where only one of the two components is optimized, and the

full complexity of the joint problem is not addressed. Our algorithms achieve scalability by orchestrating carefully designed inexact solvers for inner subproblems, for which we also provide convergence rates.

- We provide bounds on Rademacher Complexity for the vector-valued hypothesis sets considered by our algorithm. This complements and extends generalization results in the multiple kernel learning literature for the scalar case [11, 38, 19, 24, 20].
- We demonstrate that the generality of our framework enables novel non-standard applications. In particular, when applied to multivariate time series problems, sparsity in kernel combinations lends itself to a natural causality interpretation. We believe that this nonlinear generalization of graphical Granger Causality techniques (see [3, 22, 33] and references therein) may be of independent interest.

A number of results are stated in this paper for completeness, but without proofs due to space considerations. For detailed proofs, please see a version of this paper with Supplementary Material available at [34].

2 Vector-valued RLS & Separable Matrix-valued Kernels

Given labeled examples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{l}$, $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d$, $\mathbf{y}_i \in \mathcal{Y} \subset \mathbb{R}^n$, the vector-valued Regularized Least Squares (RLS) solves the following problem,

$$\underset{f \in \mathcal{H}_{\overrightarrow{k}}}{\arg\min} \frac{1}{l} \sum_{i=1}^{l} \|f(\mathbf{x}_i) - \mathbf{y}_i\|_2^2 + \lambda \|f\|_{\mathcal{H}_{\overrightarrow{k}}}^2, \qquad (1)$$

where $\mathcal{H}_{\overrightarrow{k}}$ is a vector-valued RKHS generated by the kernel function $\overrightarrow{k}$, and $\lambda > 0$ is the regularization parameter. For readers unfamiliar with vector-valued RKHS theory that is the basis of such algorithms, we provide a first-principles overview in our Supplementary Material (see [34], section 4).

In the vector-valued setting, $\overrightarrow{k}$ is a matrix-valued function, i.e., when evaluated for any pair of inputs $(\mathbf{x}, \boldsymbol{z})$, the value of $\overrightarrow{k}(\mathbf{x}, \boldsymbol{z})$ is an n-by-n matrix. More generally speaking, the kernel function is an input-dependent linear operator on the output space. The kernel function is *positive* in the sense that for any finite set of l input-output pairs $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{l}$, the following holds: $\sum_{i,j=1}^{l} \mathbf{y}_i^T \overrightarrow{k}(\mathbf{x}_i, \mathbf{x}_j)\mathbf{y}_j \geq 0$. A generalization [25] of the standard Representer Theorem says that the optimal solution has the form,

$$f(\cdot) = \sum_{i=1}^{l} \overrightarrow{k}(\mathbf{x}_i, \cdot)\boldsymbol{\alpha}_i, \qquad (2)$$

where the coefficients $\boldsymbol{\alpha}_i$ are n-dimensional vectors. For RLS, these coefficient vectors can be obtained solving a dense linear system, of the familiar form,

$$\left(\overrightarrow{\mathbf{K}} + \lambda l \mathbf{I}_{nl}\right) vec(\mathbf{C}^T) = vec(\mathbf{Y}^T),$$

where $\mathbf{C} = [\boldsymbol{\alpha}_1 \dots \boldsymbol{\alpha}_l]^T \in \mathbb{R}^{l \times n}$ assembles the coefficient vectors into a matrix; the *vec* operator stacks columns of its argument matrix into a long column vector; $\overrightarrow{\mathbf{K}}$ is a large $nl \times nl$-sized Gram matrix comprising of the blocks $\overrightarrow{k}(\mathbf{x}_i, \mathbf{x}_j)$, for $i, j = 1 \dots l$, and $\mathbf{I}_{nl}$ denotes the identity matrix of compatible size. It is easy to see that for $n = 1$, the above developments exactly collapse to familiar concepts for scalar RLS (also known as Kernel Ridge Regression). In general though, the above linear system requires $O((nl)^3)$ time to be solved using standard dense numerical linear algebra, which is clearly prohibitive. However, for a family of *separable matrix-valued kernels* [1, 9] defined below, the computational cost can be improved to $O(n^3 + l^3)$; though still costly, this is atleast comparable to scalar RLS when n is comparable to l.

Separable Matrix Valued Kernel and its Gram matrix: *Let k be a scalar kernel function on the input space $\mathcal{X}$ and $\mathbf{K}$ represent its Gram matrix on a finite sample. Let $\mathbf{L}$ be an $n \times n$ positive semi-definite output kernel matrix. Then, the function $\overrightarrow{k}(\mathbf{x}, \boldsymbol{z}) = k(\mathbf{x}, \boldsymbol{z})\mathbf{L}$ is positive and hence defines a matrix valued kernel. The Gram matrix of this kernel is $\overrightarrow{\mathbf{K}} = \mathbf{K} \otimes \mathbf{L}$ where $\otimes$ denotes Kronecker product.*

For separable kernels, the corresponding RLS dense linear system (Eqn 3 below) can be reorganized into a *Sylvester equation* (Eqn 4 below):

$$\begin{aligned} (\mathbf{K} \otimes \mathbf{L} + \lambda l \mathbf{I}_{nl})\, vec(\mathbf{C}^T) &= vec(\mathbf{Y}^T), & (3)\\ \mathbf{KCL} + \lambda l \mathbf{C} &= \mathbf{Y}. & (4) \end{aligned}$$

Sylvester solvers are more efficient than applying a direct dense linear solver for Eqn 3. The classical Bartel-Stewart and Hessenberg-Schur methods (e.g., see MATLAB's *dlyap* function) are usually used for solving Sylvester equations. They are similar in flavor to an eigendecomposition approach [1] we describe next for completeness, though they take fewer floating point operations at the same cubic order of complexity.

Eigen-decomposition based Sylvester Solver: *Let $\mathbf{K} = \mathbf{TMT}^T$ and $\mathbf{L} = \mathbf{SNS}^T$ denote the eigen-decompositions of $\mathbf{K}$ and $\mathbf{L}$ respectively, where $\mathbf{M} = diag(\sigma_1 \dots \sigma_l), \mathbf{N} = diag(\rho_1 \dots \rho_n)$. Then the solution to the matrix equation $\mathbf{KCL} + \lambda \mathbf{C} = \mathbf{Y}$ always exists when $\lambda > 0$ and is given by $\mathbf{C} = \mathbf{T}\tilde{\mathbf{X}}\mathbf{S}$ where $\tilde{\mathbf{X}}_{ij} = \frac{(\mathbf{T}^T\mathbf{YS})_{ij}}{\sigma_i \rho_j + \lambda}$.*

Output Kernel Learning: In recent work [12] develop an elegant extension of the vector-valued RLS problem (Eqn. 1), which we will briefly describe here. We will use the shorthand $\overrightarrow{k} = k\mathbf{L}$ to represent the implied separable kernel and correspondingly denote its RKHS by $\mathcal{H}_{k\mathbf{L}}$. [12] attempt to jointly learn both $f \in \mathcal{H}_{k\mathbf{L}}$ and $\mathbf{L}$, *for a fixed pre-defined choice of k*. In finite dimensional language, $\mathbf{L}$ and $\mathbf{C}$ are estimated by solving the following problem [12],

$$\underset{\mathbf{C}\in\mathbb{R}^{l\times n},\mathbf{L}\in\mathcal{S}_+^n}{\arg\min} \frac{1}{l}\|\mathbf{KCL}-\mathbf{Y}\|_F^2 + \lambda tr(\mathbf{C}^T\mathbf{KCL}) + \rho\|\mathbf{L}\|_F^2,$$

where $tr(\cdot)$ denotes trace, $\|\cdot\|_F$ denotes Frobenius norm, and $\mathcal{S}_+^n$ denotes the cone of positive semi-definite matrices. It is shown that the objective function is *invex*, i.e., its stationary points are globally optimal. [12] proposed a block coordinate descent where for fixed $\mathbf{L}$, $\mathbf{C}$ is obtained by solving Eqn. 4 using an Eigendecomposition-based solver. Under the assumption that $\mathbf{C}$ *exactly* satisfies Eqn. 4, the resulting update for $\mathbf{L}$ is then shown to automatically satisfy the constraint that $\mathbf{L} \in \mathcal{S}_+^n$. However, [12] remark that experiments on their largest dataset took roughly a day to complete on a standard desktop and that the *"limiting factor was the solution of the Sylvester equation"*.

3 Learning over a Vector-valued RKHS Dictionary

Our goals are two fold: one, we seek a fuller resolution of the separable kernel learning problem for vector-valued RLS problems; and two, we wish to derive extensible algorithms that are eigendecomposition-free and much more scalable. In this section, we expand Eqn. 1 to simultaneously learn both input and output kernels over a predefined dictionary, and develop optimization algorithms based on approximate inexact solvers that execute cheap iterations.

Consider a **dictionary of separable matrix valued kernels**, of size m, sharing the same output kernel matrix $\mathbf{L}$: $\mathcal{D}_\mathbf{L} = \{k_1\mathbf{L}, \ldots k_m\mathbf{L}\}$. Let $\mathcal{H}(\mathcal{D}_\mathbf{L})$ denote the sum space of functions:

$$\mathcal{H}(\mathcal{D}_\mathbf{L}) = \left\{ f = \sum_{j=1}^m f_j : f_j \in \mathcal{H}_{k_j\mathbf{L}} \right\}, \qquad (5)$$

and equip this space with the following l_p norms:

$$\|f\|_{l_p(\mathcal{H}(\mathcal{D}_\mathbf{L}))} = \inf_{f: f=\sum_j f_j} \left\| \left(\|f_1\|_{\mathcal{H}_{k_1\mathbf{L}}}, \ldots, \|f_m\|_{\mathcal{H}_{k_m\mathbf{L}}} \right) \right\|_p .$$

The infimum in the above definition is in fact attained as a minimum (see Proposition 2 in our Supplementary Material [34]), so that one can write

$$\|f\|_{l_p(\mathcal{H}(\mathcal{D}_\mathbf{L}))} = \min_{f: f=\sum_j f_j} \left\| \left(\|f_1\|_{\mathcal{H}_{k_1\mathbf{L}}}, \ldots, \|f_m\|_{\mathcal{H}_{k_m\mathbf{L}}} \right) \right\|_p .$$

For notational simplicity, we will denote these norms as $\|f\|_{l_p}$ though it should be kept in mind that their definition is with respect to a given dictionary of vector-valued RKHSs.

Note that $\|f\|_{l_1}$, being the l_1 norm of the vector of norms in individual RKHSs, imposes a *functional notion of sparsity* on the vector-valued function f. We now consider objective functions of the form,

$$\underset{f\in\mathcal{H}(\mathcal{D}_\mathbf{L}),\mathbf{L}\in\mathcal{S}_+^n(\tau)}{\arg\min} \frac{1}{l}\sum_{i=1}^{l} \|f(\mathbf{x}_j) - \mathbf{y}_i\|_2^2 + \lambda\Omega(f), \qquad (6)$$

where $\mathbf{L}$ is constrained to belong to the *Spectahedron* with bounded trace:

$$\mathcal{S}_+^n(\tau) = \{\mathbf{X} \in \mathcal{S}_+^n | trace(\mathbf{X}) \le \tau\},$$

where $\mathcal{S}_+^n$ denotes the cone of symmetric positive semi-definite matrices, and Ω is a regularizer whose canonical choice will be the squared l_p norm (with $1 \le p \le 2$), i.e.,

$$\Omega(f) = \|f\|^2_{l_p(\mathcal{H}(\mathcal{D}_\mathbf{L}))}.$$

When $p \to 1$, Ω induces sparsity, while for $p \to 2$, non-sparse uniform combinations approaching a simple sum of kernels is induced. Our algorithms work for a broad choice of regularizers that admit a quadratic variational representation of the form:

$$\Omega(f) = \min_{\boldsymbol{\eta}\in\mathbb{R}_+^m} \sum_{i=1}^m \frac{\|f_i\|^2_{\mathcal{H}_{k_i\mathbf{L}}}}{\eta_i} + \omega(\boldsymbol{\eta}), \qquad (7)$$

for an appropriate auxiliary function $\omega : \mathbb{R}_+^m \mapsto \mathbb{R}$. For squared l_p norms, this auxillary function is the indicator function of a convex set [5, 36, 26]:

$$\omega(\boldsymbol{\eta}) = 0 \text{ if } \eta_i \ge 0, \sum_{i=1}^m \eta_i^q \le 1, \text{ and } \infty \text{ otherwise}, \qquad (8)$$

where $q = \frac{p}{2-p} \in [1, \infty]$ for $p \in [1, 2]$.

We rationalize this framework as follows:

- ○ Penalty functions of the form above define a broad family of structured sparsity-inducing norms that have extensively been used in the multiple kernel learning and sparse modeling literature [5, 36, 27]. They allow complex non-differentiable norms to be related back to weighted l_2 or RKHS norms, and optimizing the weights $\boldsymbol{\eta}$ in many cases infact admits closed form expressions. Infact, all norms admit quadratic variational representations of related forms [5].
- ○ Optimizing $\mathbf{L}$ over the Spectahedron allows us to develop a specialized version of the approximate Sparse SDP solver [16] whose iterations involve the computation of only a single extremal eigenvector of

the (partial) gradient at the current iterate – this involves relatively cheap operations followed by quick rank-one updates.

- ◦ By bounding the trace of $\mathbf{L}$, we show below that a Conjugate Gradient (CG) based iterative Sylvester solver for Eqn. 3 would always be invoked on well-conditioned instances and hence show rapid numerical convergence (particularly also with warm starts).
- ◦ The trace constraint parameter τ, together with the regularization parameter λ, also naturally appears in our Rademacher complexity bounds.

3.1 Algorithms

First we give a basic result concerning sums of vector-valued RKHSs. The proof, given in our Supplementary Material [34], follows Section 6 of [4] replacing scalar concepts with corresponding notions from the theory of vector-valued RKHSs [25].

Proposition 1. *Given a collection of matrix-valued reproducing kernels* $\overrightarrow{k}_1 \dots \overrightarrow{k}_m$ *and positive scalars* $\eta_j > 0, j = 1 \dots m$, *the function:*

$$\overrightarrow{k}_{\boldsymbol{\eta}} = \sum_{i=1}^{m} \eta_i \overrightarrow{k}_i,$$

is the reproducing kernel of the sum space $\mathcal{H} = \{f : \mathcal{X} \mapsto \mathcal{Y} | f(\mathbf{x}) = \sum_{j=1}^{m} f_j(\mathbf{x}), f_j \in \mathcal{H}_{\overrightarrow{k}_j}\}$ *with the norm given by:*

$$\|f\|^2_{\mathcal{H}_{\overrightarrow{k}_{\boldsymbol{\eta}}}} = \min_{f=\sum_{j=1}^{m} f_i, f_j \in \mathcal{H}_{\overrightarrow{k}_j}} \sum_{j=1}^{m} \frac{\|f_j\|^2}{\eta_j}.$$

This result combined with the variational representation of the penalty function in Eqn. 7 allows us to reformulate Eqn. 6 in terms of a joint optimization problem over $\boldsymbol{\eta}, \mathbf{L}$ and $f \in \mathcal{H}_{k_{\boldsymbol{\eta}}\mathbf{L}}$, where we define the weighted scalar kernel $k_{\boldsymbol{\eta}} = \sum_{j=1}^{m} \eta_j k_j$. This formulation allows us to scale gracefully with respect to m, the number of kernels. Denote the Gram matrix of $k_{\boldsymbol{\eta}}$ on the labeled data as $\mathbf{K}_{\boldsymbol{\eta}}$, i.e., $\mathbf{K}_{\boldsymbol{\eta}} = \sum_{j=1}^{m} \eta_j \mathbf{K}_j$, where $\mathbf{K}_j$ denotes the Gram matrices of the individual scalar kernel k_j. The finite dimensional version of the reformulated problem becomes,

$$\begin{aligned} \underset{\mathbf{C}\in\mathbb{R}^{n\times l},\mathbf{L}\in\mathcal{S}_+^n(\tau),\boldsymbol{\eta}\in\mathbb{R}_+^m}{\arg\min} & \frac{1}{l}\|\mathbf{K}_{\boldsymbol{\eta}}\mathbf{C}\mathbf{L}-\mathbf{Y}\|_F^2 \\ & +\lambda\, trace\left(\mathbf{C}^T\mathbf{K}_{\boldsymbol{\eta}}\mathbf{C}\mathbf{L}\right)+\omega(\boldsymbol{\eta}). \qquad (9)\end{aligned}$$

A natural strategy for such a non-convex problem is Block Coordinate Descent. The minimization of $\mathbf{C}$ or $\mathbf{L}$ keeping the other variables fixed, is a convex optimization problem. The minimization of $\boldsymbol{\eta}$ admits closed form solution. At termination, the vector-valued function returned is

$$f^\star(\mathbf{x}) = \mathbf{L}\mathbf{C}^T[k_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{x}_1)\dots k_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{x}_l)]^T,$$

which is a matrix version of the functional form for the optimal solution as specified by the Representer theorem (Eqn. 2) for separable kernels. We next describe each of the three block minimization subproblems.

A. Conjugate Gradient Sylvester Solver: For fixed $\boldsymbol{\eta}, \mathbf{L}$, the optimal $\mathbf{C}$ is given by the solution of the dense linear system of Eqn 3 or the Sylvester equation 4, with $\mathbf{K} = \mathbf{K}_{\boldsymbol{\eta}}$. General dense linear solvers have prohibitive $O(n^3l^3)$ cost when invoked on Eqn. 3. The $O(n^3 + l^3)$ eigendecomposition-based Sylvester solver performs much better, but needs to be invoked repeatedly since $\mathbf{L}$ as well as $\mathbf{K}_{\boldsymbol{\eta}}$ are changing across (outer) iterations. Instead, we apply a CG-based iterative solver for Eqn 3. Despite the massive size of the $nl \times nl$ linear system, using CG infact has several unobvious quantifiable advantages due to the special Kronecker structure of Eqn. 3:

- ◦ A CG solver can exploit warm starts by initializing from previous $\boldsymbol{\eta}, \mathbf{L}$, and allow early termination at cheaper computational cost.
- ◦ The large $nl \times nl$ coefficient matrix in Eqn.3 never needs to be explicitly materialized. For any CG iterate $\mathbf{C}^{(k)}$, matrix-vector products can be efficiently computed since,

  $$(\mathbf{K}_\eta \otimes \mathbf{L} + \lambda l \mathbf{I}_{nl}) vec(\mathbf{C}^{(k)T}) = vec(\mathbf{K}_\eta \mathbf{C}^{(k)} \mathbf{L} + \lambda l \mathbf{C}^{(k)}).$$

  CG can exploit additional low-rank or sparsity structure in $\mathbf{K}_\eta$ and $\mathbf{L}$ for fast matrix multiplication. When the base kernels are either (a) linear kernels derived from a small group of features, or (b) arise from randomized approximations, such as the random Fourier features for Gaussian Kernel [28], then $\mathbf{K}_\eta = \sum_{j=1}^{m} \eta_j \mathbf{Z}_j \mathbf{Z}_j^T$ where $\mathbf{Z}_j$ has $d_j \ll l$ columns. In this case, $\mathbf{K}_\eta$ need never be explicitly materialized and the cost of matrix multiplication can be further reduced.

CG is expected to make rapid progress in a few iterations in the presence of strong regularization as enforced jointly by λ and the trace constraint parameter τ on $\mathbf{L}$. This is because the coefficient matrix of the linear system in Eqn. 3 is then expected to be well conditioned for all possible $\mathbf{K}_{\boldsymbol{\eta}}, \mathbf{L}$ that the algorithm may encounter, as we formalize in the following proposition. Below, let $\|\mathbf{K}_i\|_2$ denote the spectral norm of the Gram matrix $\mathbf{K}_i$, i.e., its largest eigenvalue.

Proposition 2 (Convergence Rate for CG-solver for Eqn. 3 with $\mathbf{K} = \mathbf{K}_{\boldsymbol{\eta}}$). *Assume l_1 norm for Ω in Eqn. 6. Let $\mathbf{C}^{(k)}$ be the CG iterate at step k, $\mathbf{C}^\star$ be* *the optimal solution (at current fixed $\boldsymbol{\eta}$ and $\mathbf{L}$) and $\mathbf{C}^{(0)}$ be the initial iterate (warm-started from previous value). Then,*

$$\|\mathbf{C}^{(k)} - \mathbf{C}^*\|_F \leq 2\sqrt{\phi}\left(\frac{\sqrt{\phi}-1}{\sqrt{\phi}+1}\right)^k \|\mathbf{C}^{(0)} - \mathbf{C}^*\|_F,$$

where $\phi = 1 + \frac{\gamma\tau}{l\lambda}$ with $\gamma = \max_i \|\mathbf{K}_i\|_2$. For dictionaries involving only Gaussian scalar kernels, $\phi \leq 1 + \frac{\tau}{\lambda}$, i.e., the convergence rate depends only on the relative strengths of regularization parameters λ, τ.

The proof is given in our Supplementary Material [34].

B. Updates for $\boldsymbol{\eta}$: Note from Eqn. 7 that the optimal weight vector $\boldsymbol{\eta}$ only depends on the RKHS norms of component functions, and is oblivious to the vector-valued, as opposed to scalar-valued, nature of the functions themselves. This is essentially the reason why existing results [5, 36, 26] routinely used in the (scalar) MKL literature can be immediately applied to our setting to get closed form update rules. Define $\alpha_j = \|f_j\|_{k_j\mathbf{L}} = \hat{\eta_j}\sqrt{trace(\mathbf{C}\mathbf{K}_j\mathbf{C}\mathbf{L})}$ where $\hat{\eta_j}$ refers to previous value of η_j. The components of the optimal weight vector $\boldsymbol{\eta}$ are given below for two choices of Ω.

- For $\Omega(f) = \|f\|_{l_p}^2$, the optimal $\boldsymbol{\eta}$ is given by:

$$\eta_j = \alpha_j^{\frac{2}{q+1}} / \left(\sum_{j=1}^m \alpha_j^{\frac{2}{q+1}}\right)^q \quad \text{for } q = \frac{p}{2-p} \qquad (10)$$

- For an elastic net type penalty, $\Omega(f) = (1-\mu)\|f\|_{l_1}^1 + \mu\|f\|_{l_2}^2$, we have $\eta_j = \alpha_j / (1 - \mu + \mu\alpha_j)$.

Several other choices are also infact possible, e.g., see Table 1 in [36], discussion around subquadratic norms in [5] and regularizers for structured sparsity introduced in [27].

C. Spectahedron Solver: Here, we consider the $\mathbf{L}$ optimization subproblem, which is:

$$\underset{\mathbf{L}\in\mathcal{S}_+^n(\tau)}{\arg\min}\, g(\mathbf{L}) = \frac{1}{l}\|\mathbf{A}\mathbf{L} - \mathbf{Y}\|_{fro}^2 + \lambda trace(\mathbf{B}^T\mathbf{L}), \quad (11)$$

where $\mathbf{A} = \mathbf{K}_{\boldsymbol{\eta}}\mathbf{C}$ and $\mathbf{B} = \mathbf{C}^T\mathbf{A}$. Hazan's Sparse SDP solver [16, 13] based on Frank-Wolfe algorithm [10], can be used for problems of the general form,

$$\mathbf{L}^\star = \underset{\mathbf{L}\in\mathcal{S}_+^n, trace(\mathbf{L})=1}{\arg\min} g(\mathbf{L}),$$

where g is a convex, symmetric and differentiable function. It has been successfully applied in matrix completion and collaborative filtering settings [17].

In each iteration, Hazan's algorithm optimizes a linearization of the objective function around the current iterate $\mathbf{L}^{(k)}$, resulting in updates of the form,

$$\mathbf{L}^{(k+1)} = \mathbf{L}^{(k)} + \alpha_k(\boldsymbol{v}_k\boldsymbol{v}_k^T - \mathbf{L}^{(k)}), \qquad (12)$$

where $\boldsymbol{v}_k = ApproxEV\left(\nabla g(\mathbf{L}^{(k)}), \frac{C_g}{k^2}\right)$, $\alpha_k = \min\left(1, \frac{2}{k}\right)$ and C_g is a constant which measures the curvature of the graph of g over the Spectahedron. Here, *ApproxEv* is an approximate eigensolver which when invoked on the gradient of g at the current iterate $\mathbf{L}^{(k)}$ (a positive semi-definite matrix) computes the single eigenvector corresponding to the smallest eigenvalue, only to a prespecified precision. Hazan's algorithm is appealing for us since *each iteration itself tolerates approximations* and the updates pump in rank-one terms. We specialize Hazan's algorithm to our framework as follows (below, note that $\mathbf{A} = \mathbf{K}_{\boldsymbol{\eta}}\mathbf{C}$ and $\mathbf{B} = \mathbf{C}^T\mathbf{A}$):

- Using **bounded trace constraints**, $trace(\mathbf{L}) \leq \tau$, instead of unit trace is more meaningful for our setting. The following modified updates optimize over $\mathcal{S}_+^n(\tau)$: $\mathbf{L}^{(k+1)} = \mathbf{L}^{(k)} + \alpha_k(\tau\boldsymbol{v}_k\boldsymbol{v}_k^T - \mathbf{L}^{(k)})$, where $\boldsymbol{v}_k$ is reset to the zero vector if the smallest eigenvalue is positive.
- The **gradient for our objective** is: $\nabla g(\mathbf{L}) = \boldsymbol{G} + \boldsymbol{G}^T - diag(\boldsymbol{G})$ where $\boldsymbol{G} = \lambda\mathbf{B} + 2\mathbf{A}^T\mathbf{A}\mathbf{L} - 2\mathbf{A}^T\mathbf{Y}$ and $diag(\cdot)$ assembles the diagonal entries of its argument into a diagonal matrix.
- Instead of using Hazan's line search parameter α_k, we do **exact line search** along the direction $\boldsymbol{P} = \tau\boldsymbol{v}_k\boldsymbol{v}_k - \mathbf{L}^{(k)}$ which leads to a closed form expression:

$$\alpha_k = -\frac{trace((\frac{1}{l}\mathbf{A}\mathbf{L}^{(k)} - \mathbf{Y})^T\mathbf{A}\boldsymbol{P} + \frac{1}{2}\lambda\mathbf{B}\boldsymbol{P})}{trace(\frac{1}{l}\boldsymbol{P}^T\mathbf{A}^T\mathbf{A}\boldsymbol{P})}.$$

Adapting the analysis of Hazan's algorithm in [13] to our setting, we get the following convergence rate (proof given in our Supplementary Material [34]):

Proposition 3 (Convergence Rate for optimizing $\mathbf{L}$). *Assume l_1 norm for Ω in Eqn. 6. For $k \geq 16(\tau\gamma)^2/\epsilon$, the iterate in Eqn. 12 satisfies $g(\mathbf{L}^{(k+1)}) - g(\mathbf{L}^\star) \leq \epsilon/2$ where $\gamma = \max_i \|\mathbf{K}_i\|_2$.*

Remarks: Note that Propositions 2 and 3 offer convergence rates for the convex optimization subproblems associated with optimizing $\mathbf{C}$ and $\mathbf{L}$ respectively keeping other variables fixed. These results strongly suggest that inexact solutions to subproblems may be quickly obtained in a few iterations. A full theoretical convergence analysis of Block Coordinate Descent to stationary points of the objective function under inexact updates is currently not within the scope of this paper. An empirical analysis of convergence behavior is provided in Section 4.

3.2 Rademacher Complexity Results

Here, we complement our algorithms with statistical generalization bounds. The notion of Rademacher complexity is readily generalizable to vector-valued hypothesis spaces [23]. Let $\mathcal{H}$ be a class of functions $f : \mathcal{X} \to \mathcal{Y}$, where $Y \subset \mathbb{R}^n$. Let $\boldsymbol{\sigma} \in \mathbb{R}^n$ be a vector of independent Rademacher variables, and similarly define the matrix $\Sigma = [\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_l] \in \mathbb{R}^{n\times l}$. The empirical Rademacher complexity of the vector-valued class $\mathcal{H}$ is the function $\hat{R}_l(\mathcal{H})$ defined as

$$\hat{\mathcal{R}}_l(\mathcal{H}) = \frac{1}{l}\mathbb{E}_\Sigma\left[\sup_{f\in\mathcal{H}}\sum_{i=1}^{l}\boldsymbol{\sigma}_i^T f(\mathbf{x}_i)\right]. \tag{13}$$

We now state bounds on the Rademacher complexity of hypothesis spaces considered by our algorithms, both for general matrix-valued kernel dictionaries as well as the special case of separable matrix-valued kernel dictionaries. When the output dimensionality is set to 1, our results essentially recover existing results in the *scalar* multiple kernel learning literature given in [11, 38, 19, 24]. Our bounds in part (B) and (C) in the Theorem below involve the same dependence on the number of kernels m, and on p (for l_p norms) as given in [11], though there are slight differences in stated bounds since our hypothesis class is not exactly the same as that in [11]. In particular, for the case of $p = 1$ (part C below), we obtain a $\sqrt{log\ m}$ dependence on the number of kernels which is known to be tight for the scalar case [11, 24]. Since this logarithmic dependence is rather mild, we can expect to learn effectively over a large dictionary even in the vector-valued setting.

Theorem 3.1. *Let $\mathcal{H} = \{f = \sum_{j=1}^m f_j, f_j \in \mathcal{H}_{\vec{k}_j}\}$. For $1 \le p \le \infty$, consider the hypothesis class*

$$\mathcal{H}_\lambda^p = \{f \in \mathcal{H} : f = \sum_{j=1}^m f_j, \quad f_j \in \mathcal{H}_{\vec{k}_j},$$

$$||f||_{\mathcal{H}(l_p)} = \min_{f_j\in\mathcal{H}_{\vec{k}_j}, \sum_{j=1}^m f_j = f}\left(\sum_{j=1}^m ||f_j||^p_{\mathcal{H}_{\vec{k}_j}}\right)^{1/p} \le \lambda\}.$$

(A) For any p, $1 \le p \le \infty$, the empirical Rademacher complexity of $\mathcal{H}_\lambda^p$ can be upper bounded as follows:

$$\hat{R}_l(\mathcal{H}_\lambda^p) \le \frac{\lambda\|\mathbf{u}\|_1}{l},$$

where $\mathbf{u} = \left[\sqrt{\text{trace}(\vec{\mathbf{K}}_1)}, \ldots, \sqrt{\text{trace}(\vec{\mathbf{K}}_m)}\right]$. *For the case of separable kernels, where* $\vec{k}_i(x, z) = k_i(x, z)\mathbf{L}$ *such that* $\sup_{x\in\mathcal{X}} k_i(x, x) \le \kappa$ *and* $\text{trace}(\mathbf{L}) \le \tau$, *we have*

$$\hat{R}_l(\mathcal{H}_\lambda^p) \le \lambda m\sqrt{\frac{\kappa\tau}{l}}.$$

(B) If p is such that $q \in \mathbb{N}$, where $\frac{1}{p} + \frac{1}{q} = 1$, then

$$\hat{R}_l(\mathcal{H}_\lambda^p) \le \frac{\lambda}{l}\sqrt{\eta_0 q}||\mathbf{u}||_q,$$

where $\eta_0 = \frac{23}{22}$. For separable kernels,

$$\hat{R}_l(\mathcal{H}_\lambda^p) \le \lambda m^{1/q}\sqrt{\frac{\eta_0 q\kappa\tau}{l}}.$$

(C) If $p = 1$, so that $q = \infty$, then

$$\hat{R}_l(\mathcal{H}_\lambda^1) \le \frac{\lambda}{l}\sqrt{\eta_0 r}||\mathbf{u}||_r, \forall r \in \mathbb{N}.$$

For separable kernels, we have

$$\hat{R}_l(\mathcal{H}_\lambda^1) \le \begin{cases} \lambda\sqrt{\frac{\eta_0\kappa\tau}{l}}, & \text{if } m = 1, \\ \lambda\sqrt{\frac{\eta_0 e\lceil 2\ln m\rceil\kappa\tau}{l}}, & \text{if } m > 1. \end{cases}$$

Due to space limitations, the full proof of this Theorem is provided our Supplementary Material [34].

Using well-known results [7], these bounds on Rademacher complexity can be immediately turned into generalization bounds for our algorithms.

4 Empirical Studies

Statistical Benefits of Joint Input/Output Kernel Learning: We start with a small dataset of weekly log returns of 9 stocks from 2004, studied in [39, 31] in the context of linear multivariate regression with output covariance estimation techniques. We consider first-order vector autoregressive (VAR) models of the form $\mathbf{x}_t = f(\mathbf{x}_{t-1})$ where $\mathbf{x}_t$ corresponds to the 9-dimensional vector of log-returns for the 9 companies at week t and the function f is estimated by solving Eqn. 6. Our experimental prototcol is exactly the same as [39, 31]: data is split evenly into a training and a test set and the regularizaton parameter λ is chosen by 10-fold cross-validation. All other parameters are left at their default values (i.e., $p = 1$). We generated a dictionary of 117 Gaussian kernels defined by univariate Gaussian kernels on each of the 9 dimensions with 13 varying bandwidths. Results are shown in Table 1 where we compare our methods in terms of mean test RMSE against standard linear regression (OLS) and linear Lasso independently applied to each output coordinate, and the sparse multivariate regression with covariance estimation approaches of [31, 39], labeled MRCE and FES respectively. *We see that joint input and output kernel learning (labeled IOKL) yields the best return prediction model reported to date on this dataset.* As expected, it outperforms models obtained by leaving output kernel matrix fixed as the identity

Table 1: VAR modeling on financial datasets.

	OLS	Lasso	MRCE	FES	IKL	OKL	IOKL
WMT	0.98	0.42	0.41	0.40	0.43	0.43	0.44
XOM	0.39	0.31	0.31	0.29	0.32	0.31	0.29
GM	1.68	0.71	0.71	0.62	0.62	0.59	**0.47**
Ford	2.15	0.77	0.77	0.69	0.56	0.48	**0.36**
GE	0.58	0.45	0.45	0.41	0.41	0.40	**0.37**
COP	0.98	0.79	0.79	0.79	0.81	0.80	**0.76**
Ctgrp	0.65	0.66	0.62	0.59	0.66	0.62	**0.58**
IBM	0.62	0.49	0.49	0.51	0.47	0.50	**0.42**
AIG	1.93	1.88	1.88	1.74	1.94	1.87	1.79
Average	1.11	0.72	0.71	0.67	0.69	0.67	**0.61**

and only optimizing scalar kernels (IKL), or only optimizing the output kernel for fixed choices of scalar kernel (OKL). Of the 117 kernels, 13 have 97% of the mass in the learnt scalar kernel combination.

Scalability and Numerical Behaviour: Our main interest here is to observe the classic tradeoff in numerical optimization between running few, but very expensive steps versus executing several cheap iterations. We use a 102-class image categorization dataset – Caltech-101 – which has been very well studied in the multiple kernel learning literature [12, 37, 14]. There are 30 training images per category for a total of 3060 training images, and 1355 test images. Targets are 102-dimensional class indicator vectors. We define a dictionary of kernels using 10 scalar-valued kernels precomputed from visual features and made publically available by the authors of [37], for 3 training/test splits. From previous studies, it is well known that all underlying visual features contribute to object discrimination on this dataset and hence non-sparse multiple kernel learning with $l_p, p > 1$ norms are more effective. We therefore set $p = 1.7$ and $\lambda = 0.001$ without any further tuning, since their choice is not central to our main goals in this experiment. We vary the stopping criteria for our CG-based Sylvester solver (cg_ϵ) and the number of iterations (sdp_{iter}) allowed in the Sparse SDP solver, for the $\mathbf{C}$ and $\mathbf{L}$ subproblems respectively. Note that the closed form $\boldsymbol{\eta}$ updates (Eqn. 10) for l_p norms take negligible time.

We compare our algorithms with an implementation in which each subproblem is solved exactly using an eigendecomposition based Sylvester solver for $\mathbf{C}$, and unconstrained updates for $\mathbf{L}$ developed in [12], respectively. To make comparisons meaningful, we set τ to a large value so that the optimization over $\mathbf{L} \in \mathcal{S}^n_+(\tau)$ effectively corresponds to unconstrained minimization over the entire psd cone $\mathcal{S}^n_+$. In Figure 1, 2, we report the improvement in objective function and classification accuracy as a function of time (upto 1 hour). We see that insufficient progress is made in both extremes: when either the degree of inexactness is intolerable ($cg_\epsilon = 0.1, sdp_{iter} = 100$) or when subproblems are solved to very high precision ($cg_\epsilon = 10^{-6}, sdp_{iter} = 3000$). *Our solvers are far more efficient than eigendecomposition based implementation that takes an exorbitant amount of time per iteration for exact solutions. Approximate solvers at appropriate precision (e.g., $cg_\epsilon = 0.01, sdp_{iter} = 1000$) make very rapid progress and return high accuracy models in just a few minutes.* In fact, averaged over the three training/test splits, *the classification accuracy obtained is* $79.43\% \pm 0.67$ *which is highly competitive with state of the art results reported on this dataset*, with the kernels used above. For example, [37] report $78.2\% \pm 0.4$, [14] report $77.7\% \pm 0.3$ and [12] report 75.36%.

Figure 1: Objective function vs time

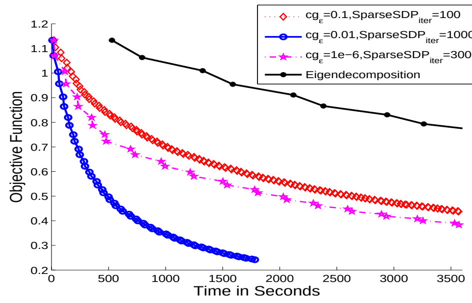
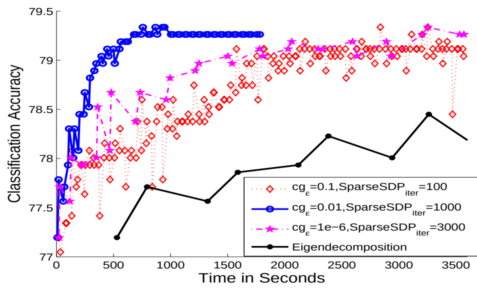


Figure 2: Accuracy vs time



4.1 Application: Non-linear Causal Inference

Here, our goal is to show how high-dimensional causal inference tasks can be naturally cast as sparse function estimation problems within our framework, leading to novel nonlinear extensions of Grouped Graphical Granger Causality techniques (see [33, 22] and references therein). In this setting, there is an interconnected system of N distinct sources of high dimensional time series data which we denote as $\mathbf{x}^i_t \in \mathbb{R}^{d_i}, i = 1 \dots N$. We refer to these sources as "nodes". The system is observed from time $t = 1$ to $t = T$, and the goal is to infer the causal relationships between the nodes. Let G denote the adjacency matrix of the unknown causal interaction graph where $G_{ij} > 0$ implies that node i causally influences node j. In 1980, Clive Granger gave an operational definition for Causality:

Granger Causality [15]: *A subset of nodes $A_i = \{j : G_{ij} > 0\}$ is said to causally influence node i, if the past values of the time series collectively associated with the node subset A_i is predictive of the future evolution of the time series associated with node i, with statistical significance, and more so than the past values of i alone.*

A practical appeal of this definition is that it links causal inference to prediction, with the caveat that causality insights are bounded by the quality of the underlying predictive model. Furthermore, the prior knowledge that the underlying causal interactions are highly selective makes sparsity a meaningful prior to use. Prior work on using sparse modeling techniques to uncover causal graphs has focused on linear models [33, 22] while many, if not most, natural systems involve nonlinear interactions for which a functional notion of sparsity is more appropriate.

To apply our framework to such problems, we model the system as the problem of estimating N nonlinear functions: $\mathbf{x}_t^i = f^i\left(\mathbf{x}_t^1, \mathbf{x}_{t-1}^1 \ldots \mathbf{x}_{t-L}^1, \ldots, \mathbf{x}_t^N, \mathbf{x}_{t-1}^N \ldots \mathbf{x}_{t-L}^N\right)$, for $1 \le i \le N$, and where L is a lag parameter. The dynamics of each node, f^i, can be expressed as the sum of a set of vector-valued functions,

$$f^i = \sum_{j=1,s}^{N} f_{j,s}^i \qquad (14)$$

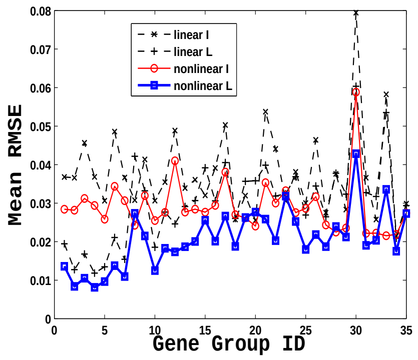
where the component $f_{j,s}^i$, for all values of the index s, *only* depends on the history of node j, i.e., the observations $\mathbf{x}_{t-1}^j \ldots \mathbf{x}_{t-L}^j$. Each $f_{j,s}^i$ belongs to vector-valued RKHS whose kernel is $k_{j,s}(\cdot,\cdot)\mathbf{L}^i$. In other words, we set up a dictionary of separable matrix-valued kernels $\mathcal{D}_{\mathbf{L}^i} = \{k_{j,s}\mathbf{L}^i\}_{j,s}$, where scalar kernels $k_{j,s}$ depend only on individual nodes j alone; and the output matrix $\mathbf{L}^i$ is associated with node i currently being modeled. By imposing (functional) sparsity in the sum in Eqn. 14 using our framework, i.e. estimating f^i by solving Eqn. 6, we can identify which subset of nodes are causal drivers (in the Granger sense) of the dynamics observed at node i. The sparsity structure of f^i then naturally induces a weighted causal graph G:

$$G_{ij} = \sum_s \eta_{j,s}^i$$

where $\eta_{j,s}^i$ are the kernel weights estimated by our algorithm. Note that $G_{ij} \neq 0$ only if a component function associated with the history of node j (for some s) is non-zero in the sum Eqn. 14). In addition to recovering the temporal causal interactions in this way, the estimated output kernel matrix $\mathbf{L}^i$ associated with each f^i captures within-source temporal dependencies. We now apply these ideas to a problem in computational biology.

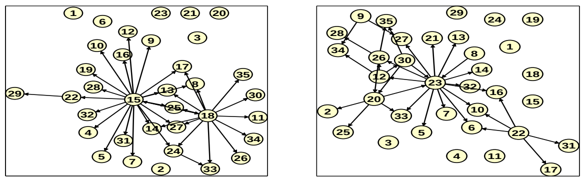
Causal Inference of Gene Networks: We use time-course gene expression microarray data measured during the full life cycle of Drosophila melanogaster [2]. The expression levels of 4028 genes are simultaneously measured at 66 time points corresponding to various developmental stages. We extracted time series data for 2397 unique genes, and grouped them into 35 functional groups based on their gene ontologies. The goal is to infer causal interactions between functional groups (represented by multiple time series associated with genes in that group), as well obtain insight on within-group relationships between genes. We conducted four sets of experiments: with linear and nonlinear dictionaries (Gaussian kernels with 13 choices of bandwidths per group), and with or without output kernel learning. We use the parameters $\lambda = 0.001$ and time lag of 7 without tuning. Figure 3 shows hold-out RMSE from the four experiments, for each of the 35 functional groups. Clearly, nonlinear models with both input and output kernel learning (labeled "nonlinear L" in Figure 3) give the best predictive performance implying greater relability in the implied causal graphs.

Figure 3: RMSE in predicing multiple time series



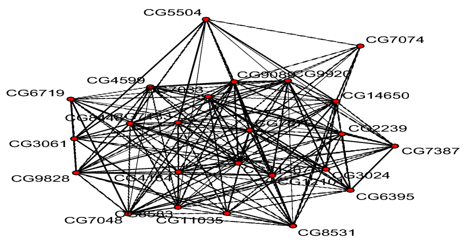
In consultation with a professional biologist, we analyzed the causal graphs uncovered by our approach (Figure 4). In particular the nonlinear causal model uncovered the centrality of a key cellular enzymatic activity, that of helicase, which was not recognized by the linear model. In contrast, the central nodes in the linear model are related to membranes (lipid binding and gtpase activity). Nucleic acid binding transcription factor activity and transcription factor binding are both related to the helicase activity, which is consistent with biological knowledge of them being tightly coupled. This was not captured in the linear model. Molecular chaperone functions, which connect ATPase activity and unfolded protein binding, was successfully identified by our model, while the linear model failed to recognize its relevance. It is less likely that unfolded protein and lipid activity should be linked as suggested by the linear model.

Figure 4: Causal Graphs: Linear (left) and Non-linear (right)



In addition, via output kernel matrix estimation (i.e., $\mathbf{L}^i$), our model also provides insight on the conditional dependencies within genes, shown in Figure 5, for the *unfolded protein binding* group.

Figure 5: Interactions in *unfolded protein binding* group



5 Related work and Conclusion

Our work is the first to address efficient simultaneous estimation of both the input and output components of separable matrix-valued kernels. Two recent papers are closely related. In [12], the input scalar kernel is predefined and held fixed, while the output matrix is optimized in a block coordinate descent procedure. As discussed in Section 2, this approach involves solving Sylvester equations using eigendecomposition methods which is computationally very costly. In very recent work, concurrent with our work, [18] independently propose a multiple kernel learning framework for operator-valued kernels. Some elements of their work are similar to ours. However, they only optimize the scalar input kernel keeping the output matrix fixed. Their optimization strategy also includes eigendecomposition, and Gauss-Siedel iterations for solving linear systems, while we exploit the quadratic nature of the objective function using CG and a fast sparse SDP solver to demonstrate the scalability benefits of inexact optimization. In addition, we provide generalization analysis in terms of bounds on the Rademacher complexity of our vector-valued hypothesis spaces, complementing analogous results in the scalar multiple kernel learning literature [11, 20]. We also outlined how our framework operationalizes nonlinear Granger Causality in high-dimensional time series modeling problems, which may be of independent interest. Future work includes extending our framework to other classes of vector-valued kernels [1, 9] and to functional data analysis problems [30].

Acknowledgments: We thank Haim Avron, Satyen Kale, Rick Lawrence, Bonnie Ray and Tara Sainath for several technically insightful and enthusiastic conversations on this work.

References

[1] M. A. Alvarez, L. Rosasco, and N. Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends in Machine Learning*, 4(3):195–266, 2012.

[2] M. Arbeitman, E. Furlong, F. Imam, E. Johnson, B. Null, B. Baker, M. Krasnow, M. Scott, R. Davis, and K. White. Gene expression during the life cycle of drosophila melanogaster., 2002.

[3] A. Arnold, Y. Liu, and N. Abe. Temporal causal modeling with graphical granger methods. In *KDD*, 2007.

[4] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

[5] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity inducing penalties. *Foundations and Trends in Machine Learning*, 2011.

[6] G. Bakir, T. Hofmann, B. Schlkopf, A. Smola, B. Taskar, and S. E. Vishwanathan. *Predicting Structured Data*. MIT Press, 2007.

[7] P. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *JMLR*, 3:463–482, 2002.

[8] P. Buhlmann and S. V. D. Geer. *Statistics for High Dimensional Data*. Springer, 2010.

[9] A. Caponnetto, M. Pontil, C.Micchelli, and Y. Ying. Universal multi-task kernels. *Journal of Machine Learning Research*, 9:1615–1646, 2008.

[10] K. Clarkson. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Transactions on Algorithms*, 2010.

[11] C. Cortes, M. Mohri, and A. Rostamizadeh. Generalization bounds for learning kernels. In *ICML*, 2010.

[12] P. G. G. P. Francesco Dinuzzo, Cheng Soon Ong. Learning output kernels with block coordinate descent. In *ICML*, 2011.

[13] B. Gartner and J. Matousek. *Approximation Algorithms and Semi-definite Programming*. Springer-Verlag, Berlin Heidelberg, 2012.

[14] P. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *ICCV*, 2009.

[15] C. Granger. Testing for causality: A personal viewpoint. *Journal of Economic Dynamics and Control*, 2:329–352, 1980.

[16] E. Hazan. Sparse approximate solutions to semidefinite programs. In *LATIN*, 2008.

[17] M. Jaggi and M. Sulovsky. A simple algorithm for nuclear norm regularized problems. In *ICML*, 2010.

[18] H. Kadri, A. Rakotomamonjy, F. Bach, and P. Preux. Multiple operator-valued kernel learning. In *NIPS*, 2012.

[19] S. Kakade, S. Shalev-Schwartz, and A. Tewari. Regularization techniques for learning with matrices. In *Journal of Machine Learning Research*, 2012.

[20] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. l_p-norm multiple kernel learning. *JMLR*, 12:953–997, 2011.

[21] V. Koltchinskii and M. Yuan. Sparsity in multiple kernel learning. *The Annals of Statistics*, 38(6):3660–3695, 2010.

[22] A. C. Lozano, N. Abe, Y. Liu, and S. Rosset. Grouped graphical granger modeling methods for temporal causal modeling. In *KDD*, pages 577–586, 2009.

[23] A. Maurer. The rademacher complexity of linear transformation classes. In *Proceedings of the Conference on Learning Theory (COLT)*, 2006.

[24] A. Maurer and M. Pontil. Structured sparsity and generalization. *Journal of Machine Learning Research*, 13:671–690, 2012.

[25] C. A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17:177–204, 2005.

[26] C. Michelli and M. Pontil. Learning the kernel function via regularization. *JMLR*, 6:1099–1125, 2005.

[27] C. Michelli and M. Pontil. Regularizers for structured sparsity. *Advances in Computational Mathematics*, 2011.

[28] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.

[29] A. Rakotomamonjy, F.Bach, S. Cano, and Y. Grandvalet. Simplemkl. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

[30] J. Ramsay and B. W. Silverman. *Functional Data Analysis*. Springer, Providence, RI, 2005.

[31] A. J. Rothman, E. Levina, and J. Zhu. Sparse multivariate regression with covariance estimation. *Journal of Computational and Graphical Statistics*, 1:947–962, 2010.

[32] L. Schwartz. Sous-espaces hilbertiens d'espaces vectoriels topologiques et noyaux associés (noyaux reproduisants). *J. Analyse Math.*, 13:115–256, 1964.

[33] A. Shojaie and G. Michailidis. Discovering graphical granger causality using the truncating lasso penalty. *Bioinformatics*, 26(18):i517–i523, Sept. 2010.

[34] V. Sindhwani, H. Q. Minh, and A. Lozano. Supplementary material at: `http://arxiv.org/abs/1210.4792v2`. Technical report, 2013.

[35] A. Tikhonov. Regularization of incorrectly posed problems. *Sov. Math. Dokl.*, 17:4:1035–1038, 1963.

[36] R. Tomioka and T. Suzuki. Regularization strategies and empirical bayesian learning for mkl. In *NIPS Workshops 2010*, 2010.

[37] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *International Conference on Computer Vision*, 2009.

[38] Y. Ying and C. Cambell. Generalization bounds for learning the kernel problem. In *COLT*, 2009.

[39] M. Yuan, A. Ekici, Z. Lu, and R. Monteiro. Dimension reduction and coefficient estimation in multivariate linear regression. *Journal of the Royal Statistical Society Series B*, 69:329–46, 2007.

Preference Elicitation For General Random Utility Models

Hossein Azari Soufiani
SEAS[*], Harvard University
azari@fas.harvard.edu

David C. Parkes
SEAS, Harvard University
parkes@eecs.harvard.edu

Lirong Xia
SEAS, Harvard University
lxia@seas.harvard.edu

Abstract

This paper discusses *General Random Utility Models (GRUMs)*. These are a class of parametric models that generate partial ranks over alternatives given attributes of agents and alternatives. We propose two preference elicitation scheme for GRUMs developed from principles in Bayesian experimental design, one for social choice and the other for personalized choice. We couple this with a general Monte-Carlo-Expectation-Maximization (MC-EM) based algorithm for MAP inference under GRUMs. We also prove uni-modality of the likelihood functions for a class of GRUMs. We examine the performance of various criteria by experimental studies, which show that the proposed elicitation scheme increases the precision of estimation.

1 Introduction

In many situations, we need to know the preferences of agents over a set of alternatives, in order to make decisions. For example, in recommender systems, we can compute recommendations of new products for a user based on his reported preferences over some products. In social choice, we need to know agent preferences over alternatives, to make a joint decision. Predicting consumer behavior based on reported preferences is an important topic in econometrics [3, 4].

There are two closely related challenges in building a decision support system: preference acquisition and computer-aided decision making [7].

Given preferences, the decision making problem can typically be solved through optimization techniques (e.g., computing the choice that minimizes the maximum *regret*). However, there is often a *preference bottleneck*, where it is too costly or even impossible for users to report full information about their preferences. This happens, for example, in airline recommendation systems, where the number of possible itineraries is large [7]. Another instance is combinatorial voting, where agents vote on multiple related issues [15].

To overcome the preference bottleneck, a well accepted approach is *preference elicitation*. This aims to elicit as little as possible of the agents' preferences, to make a good decision. Previous work focused on achieving one of the following two goals:

1. Social choice. We want to make a joint decision for all agents. Applications include combinatorial auctions [21], voting [11, 17], and crowdsourcing [19].

2. Personalized choice. We want to "learn" an agent's preferences based on a part of her own preferences or preferences of other similar agents. Applications include product configuration [7]. See [5, 13] for recent developments.

In this paper, we focus on elicitation for *ordinal preference*, which means that the agents' preferences are represented by rankings. We assume that preferences are generated by *general random utility models (GRUMs)*. In a GRUM, an agent's preferences are generated as follows: Each alternative is characterized by a *utility distribution*, and the agents rank the alternatives according to the *perceived utilities*, which are generated from the corresponding utility distributions. Parameters for each utility distribution are computed by a combination of attributes of the alternative and attributes of the agent. Parameters of the GRUM model the interrelationship between alternative attributes and agent attributes. See Section 2.1 for more details.

GRUMs are a significant extension of *random utility models (RUMs)* [22], where the effect of attributes of alternatives and agents are not considered. RUMs have been extensively studied and applied in prior work but generally in ways that are specialized to particular parametric forms; e.g., the Bradley-Terry model [8] and the Plackett-Luce model [18, 20].

[*]School of Engineering and Applied Sciences.

1.1 Contributions

We propose a general adaptive method (Algorithm 1) for preference elicitation within the *Bayesian experimental design* framework (see, [10, e.g.]), guided by maximum expected information gain. In this paper, we focus on a special case, where in each step a targeted agent reports her preferences in full.

We target an agent for elicitation who, based on agent attributes, will provide the greatest expected information gain. In addition to using classical criteria in Bayesian experimental design, we also propose two new criteria that are designed to best improve the quality of the inferred rank preferences, one for predicting social choice, and the other for predicting personalized choice.

Directly computing the optimal agent to target next can be challenging due to the lack of efficient algorithms for MAP inference and lack of efficient computation of observed Fisher information. To overcome this, we extend the MC-EM algorithm and conditions for convergence developed for RUMs by Azari et al. [1] to handle GRUMs. We compute observed Fisher information within the E-step.

We test the prosed methods for MAP/MLE inference and preference elicitation for GRUMs on both synthetic dataset and the Sushi dataset [14].

We compare the performance under the new criteria and performance under the standard criteria from Bayesian experimental design literature. Results show that our elicitation framework can significantly improve the precision of estimation for a moderate number of samples in social choice, relative to random and some of the classical elicitation criteria.

1.2 Related Work

GRUMs are a specific case of the generative model studied by Berry, Levinsohn and Pakes (therefore BLP) [3]. The BLP model explicitly considers unobserved attributes of alternatives and agents, whereas GRUMs only consider observed attributes.

However, most work on the BLP model has focused on calculating aggregate properties (for example, the demand curve) when a distribution of the values of unobserved attributes are given. Moreover, the methodologies developed in [3] and subsequent papers only work for the *logit model*. That is: the utility distributions are the standard Gumbel distribution, which is a special case. Even when there are no unobserved variables, BLP was not known to be computationally tractable, beyond the logit case.

An approximate method, that of maximum simulated likelihood has been proposed for GRUMs [23]. We focus on MAP/MLE inference and preference elicitation for GRUMs. We developed an MC-EM algorithm for a large class of GRUMs. To the best of our knowledge, this is the first practical algorithm for MAP/MLE inference for general GRUMs, beyond the logit case. We note that RUMs are a special class of GRUMs. Therefore, the new algorithm naturally extends the algorithm developed by Azari et al. [1] for RUMs. [1]

For social choice, the elicitation scheme designed by Lu and Boutilier [17] aims at computing the outcomes of different commonly studied voting rules. In comparison, the proposed elicitation scheme aims at computing the MAP of GRUMs, which we believe to be different from any commonly studied voting rules.

Compared to the elicitation scheme designed by Pfeiffer et al. [19], which adopted the *Bradley-Terry model*, this paper focuses on GRUMs, which is much more general. Also, as we will see later in the paper in Example 2, the elicitation scheme by Pfeiffer et al. is closely related to a well studied criterion under the Bayesian experimental design framework called *D-optimality*. In contrast, the new elicitation framework allows us to use many other classical criteria in Bayesian experimental design, including D-optimality. Moreover, surprisingly, experimental results on synthetic data show that D-optimality might not be a good choice for social choice for rankings.

The new elicitation framework considers the attributes of agents and alternatives, allowing for more options for elicitation (e.g. we can target an agent with specific attributes). The proposed method is related to the general idea in [13, 9, 6]. However, the proposed method is more general, in the sense that we can handle orders with any length (e.g. Sushi dataset which includes full orders and not only pairwise data). It can also handle any partial order situation due to missing data or design of voting rule (e.g. k first voting or ranks for some missing parties).

2 Preliminaries

In this section, we formally define GRUMs and their corresponding MAP mechanism. Further, we recall basic ideas in Bayesian experimental design.

2.1 General Random Utility Models

We consider a preference aggregation setting with a set of alternatives $\mathcal{C} = \{c_1, .., c_m\}$, and multiple agents indexed by $i \in \{1, \dots, n\}$. In GRUMs, for every $j \leq m$, alternative j is characterized by a vector of $L \in \mathbb{M}$ real numbers, denoted by $\vec{z}_j$. And for every $i \leq n$, agent i is characterized by a vector of $K \in \mathbb{N}$ real numbers, denoted by $\vec{x}_i$.[2]

[1]Inference and elicitation for GRUMs with unobserved attributes are two interesting directions for future research.

[2]In this paper we focus on the case where all $\vec{x}_i$ and $\vec{z}_j$ are numerical attributes rather than categorical attributes.

Throughout the paper, j denotes an alternative, i denotes an agent, l denotes the attribute of an alternative, and k denotes an agent attribute.

The agents' preferences are generated through the following process.[3] Let u_{ij} be agent i's *perceived utility* for alternative j, and let B be a $K \times L$ real matrix that models the linear inter-relation between attributes of alternatives and attributes of agents.

$$u_{ij} = \delta_j + \vec{x}_i B(\vec{z}_j)^T + \epsilon_{ij}, \quad (1)$$
$$u_{ij} \sim \Pr(\cdot|\vec{x}_i, \vec{z}_j, \delta_j, B) \quad (2)$$

In words, agent i's utility for alternative j is composed of the following three parts:

1. δ_j: The *intrinsic utility* of alternative j, which is the same across all agents;
2. $\vec{x}_i B(\vec{z}_j)^T$: The *agent-specific utility*, where B is the same across all agents;
3. ϵ_{ij}: The random noise generated independently across agents and alternatives.

Given this, an agent ranks the alternatives according to her perceived utilities for the alternatives in the descending order. That is, for agent i, $c_{j_1} \succ_i c_{j_2}$ if and only if $u_{ij_1} > u_{ij_2}$.[4] The parameters for a GRUM are denoted by $\Theta = (\vec{\delta}, B)$. When $K = L = 0$, the GRUM model degenerates to RUM.

Example 1 Figure 1 illustrates a GRUM for three alternatives (different kinds of sushi) and n agents. Each alternative is characterized by its attributes including heaviness, price, and custom loyalty. Each agent is characterized by attributes including gender and age. Agent attributes have different relationships with alternative attributes. For instance, a person's salary can be related to a preference in regard to the sushi's price rather than heaviness. The outcome of this relationship is a vector of nondeterministic utilities, assigned to the alternatives by each agent.

2.2 MAP Inference

Given a GRUM, the preference profile is viewed as *data*, $D = \{\pi^1, \ldots, \pi^n\}$, where each π^i is a permutation $(\pi^i(1), \ldots, \pi^i(m))$ of $\{1, \ldots, m\}$ that represents the full ranking $[c_{\pi^i(1)} \succ_i c_{\pi^i(2)} \succ_i \cdots \succ_i c_{\pi^i(m)}]$. We take the standard maximum a posteriori (MAP) approach to estimate the parameters.

Recall that each agent's preferences are generated conditionally independently given the parameters Θ. Therefore, in GRUMs, the probability (likelihood) of the data given

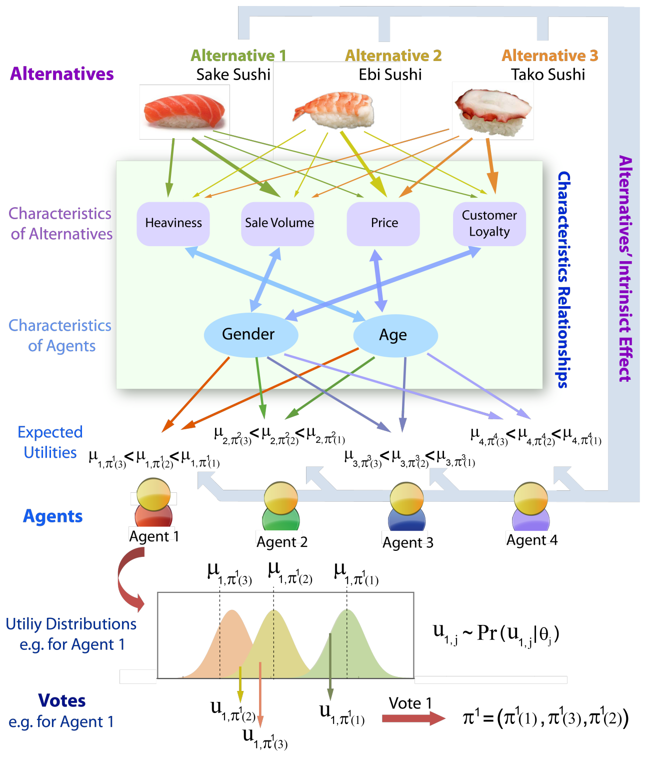

Alternatives
Alternative 1
Sake Sushi
Alternative 2
Ebi Sushi
Alternative 3
Tako Sushi
Characteristics of Alternatives
Heaviness
Sale Volume
Price
Customer Loyalty
Characteristics Relationships
Alternatives' Intrinsict Effect
Characteristics of Agents
Gender
Age
Expected Utilities
$\mu_{1,\pi^1(3)} < \mu_{1,\pi^1(2)} < \mu_{1,\pi^1(1)}$
$\mu_{2,\pi^2(3)} < \mu_{2,\pi^2(2)} < \mu_{2,\pi^2(1)}$
$\mu_{3,\pi^3(3)} < \mu_{3,\pi^3(2)} < \mu_{3,\pi^3(1)}$
$\mu_{4,\pi^4(3)} < \mu_{4,\pi^4(2)} < \mu_{4,\pi^4(1)}$
Agents
Agent 1
Agent 2
Agent 3
Agent 4
Utiliy Distributions e.g. for Agent 1
$\mu_{1,\pi^1(3)}$
$\mu_{1,\pi^1(2)}$
$\mu_{1,\pi^1(1)}$
$u_{1,j} \sim \Pr(u_{1,j}|\theta_j)$
Votes e.g. for Agent 1
$u_{1,\pi^1(2)}$
$u_{1,\pi^1(3)}$
$u_{1,\pi^1(1)}$
Vote 1
$\pi^1 = (\pi^1(1), \pi^1(3), \pi^1(2))$

Figure 1: The generative process for GRUMs.

the ground truth Θ is: $\Pr(D \mid \Theta) = \prod_{i=1}^n \Pr(\pi^i \mid \Theta)$, where:

$$\Pr(\pi^i|\Theta) = \int_{u_{i\pi^i(1)} > \cdots > u_{i\pi^i(n)}} \prod_j \Pr(u_{i\pi^i(j)}|\vec{x}_i, \vec{z}_j, \Theta)\, du_{i\pi^i(j)}$$

Suppose we have a prior over the parameters, for MAP inference we aim at computing Θ to maximize the posterior function:

$$\Pr(\Theta|D) = \prod_{i=1}^n \Pr(\pi^i \mid \Theta) \Pr(\Theta)$$

After computing Θ^* that maximizes posterior, we can make joint decisions for the agents based on Θ^*.[5] For example, we can choose the winner to be the alternative whose utility distribution has the highest mean, or choose a winning ranking over alternatives by ranking the means of the utility distributions.

2.3 One-Step Bayesian Experimental Design

Suppose we have a parametric probabilistic model. Let $\Pr(\Theta^*)$ denote the prior distribution over the parameters. A one-step Bayesian experimental design problem is composed of two parts: a set of *designs* $\mathscr{H}$ and a *quality func-*

[3]For better presentation, throughout the paper we assume that the preferences are full rankings. The results and algorithms can be extended to the case where the preferences are partial rankings.

[4]For all reasonable GRUMs the situations with tied perceived utilities have zero probability measure.

[5]In the context of social choice, the prior is often uniform, and MAP becomes MLE.

tion $G(\cdot)$ defined on any *distribution* over the parametric space.

A design $h \in \mathscr{H}$ is mathematically characterized by $\Pr(\cdot|\Theta^*, h)$ that controls the way the data D are generated for any ground truth parameter vector Θ^*. Therefore, for any given design h, we can compute the probability for data D as $\Pr(D|h)$. Given any data D and design h, we can compute the posterior distribution of parameters $\Pr(\cdot|D, h)$. The objective of Bayesian experimental design is to choose the design h that maximizes the expected quality of the posterior of MAP parameters, where the randomness comes from the data that are generated given h. Formally, we aim at computing h^* as follows.

$$h^* = \arg\max_h \int G(\Pr(\cdot|D,h)) \times \Pr(D|h)\, dD \qquad (3)$$

Often, directly computing (3) is hard. Even $G(\Pr(\cdot|D, h))$ is difficult to compute given D and h. Researchers have taken various approximations to (3). A common approach is to approximate $\Pr(\cdot|D, h)$ by a normal distribution $\mathcal{N}(\hat{\Theta}, [R(\hat{\Theta}) + I_h(\hat{\Theta})]^{-1})$, where:

- $\hat{\Theta}$ is the MAP of D,
- $R(\Theta)$ is the *precision matrix* of the prior over Θ, that is, $R = \nabla^2_\Theta \log \Pr(\Theta)$, and
- $I_h(\hat{\Theta})$ is the *Fisher information* matrix defined as follows. Let $X_\pi = \nabla_\Theta \log \Pr(\pi|\vec{\Theta}, h)$, we have

  $$I_h(\hat{\Theta}) = E_\pi(X_\pi (X_\pi)^T|_{\Theta=\hat{\Theta}}).$$

  Equivalently, if $\log \Pr(\pi|\Theta, h)$ is twice differentiable w.r.t. Θ for each ranking π, then

  $$I_h(\hat{\Theta}) = -E_\pi(\nabla^2_\Theta \log \Pr(\pi|\Theta,h)|_{\Theta=\hat{\Theta}}).$$

If we approximate $\Pr(\cdot|D, h)$ by $\mathcal{N}(\hat{\Theta}, [R(\hat{\Theta}) + I_h(\hat{\Theta})]^{-1})$, then the most commonly studied quality functions are functions of $\hat{\Theta}$ and h. More precisely, they are functions of $\hat{\Theta}$ and $R(\hat{\Theta}) + I_h(\hat{\Theta})$. In such cases, we can rewrite $G(\mathcal{N}(\hat{\Theta}, I_h(\hat{\Theta}))) = G^*_R(\hat{\Theta}, h)$. Then, (3) becomes:

$$h^* = \arg\max_h \int G^*_R(\hat{\Theta}, h) \cdot \Pr(\hat{\Theta}|h) d\hat{\Theta} \qquad (4)$$

Still the integration in (4) is often hard to compute, and is approximated by $G^*_R(\Theta^*, h)$, where Θ^* is the mode of $\Pr(\Theta)$. Some popular quality functions and corresponding approximations are summarized in Table 1.

Example 2 *The adaptive elicitation approach by Pfeiffer et al. [19] is a special case of Bayesian D-optimality design, where $\mathscr{H}$ is the set of all pairwise questions between alternatives. Pfeiffer et al. derived formulas for $\Pr(\cdot|\Theta^*, h)$ for each $h \in \mathscr{H}$, and chose h^* according to (3). The quality function they use is the negative Shannon entropy, which is exactly D-optimality as shown in Table 1.*

3 Our Preference Elicitation Scheme

In the new elicitation framework, we adapt the one-step Bayesian experimental design to multiple iterations. For any iteration t, let D^t denote the preferences elicited in all previous iterations. The prior distribution $\Pr^t$ over parameters is the posterior of observing D^t, that is: for any Θ, $\Pr^t(\Theta) = \Pr(\Theta|D^t)$. Then we solve a standard one-step Bayesian experimental design problem w.r.t. the prior $\Pr^t$ to elicit a new agents' preferences, and then form D^{t+1} for the next iteration.

Our general elicitation framework for GRUMs is presented as Algorithm 1. To allow flexibility of using various criteria of Bayesian experimental design, we let the input consist of the heuristic $G^*_R(\hat{\Theta}, h)$, which is usually a function of $\hat{\Theta}$ and $R(\hat{\Theta}) + I_h(\hat{\Theta})$. To present the main idea, in this paper the set of designs $\mathscr{H}$ is the multi-set of all agents attributes. That is, in each iteration (Steps 1∼3) we will compute an $h \in \mathscr{H}$ and query the preferences of a random agent whose attributes are h.[6] Steps 1∼3 are hard to compute. In this paper, we will use a multivariate normal distribution $\mathcal{N}(\hat{\Theta}, J_{D^t}(\hat{\Theta})^{-1})$ to approximate $\Pr(\Theta|D^t)$ in Step 2, where $J_{D^t}(\hat{\Theta})$ is the *observed Fisher information* matrix, and we immediately have $R^t = J_{D^t}(\hat{\Theta})$.[7] Given any data D, $J_D(\hat{\Theta}, h)$ is defined as follows. Again, let $\hat{\Theta} = \text{MAP}(D)$.

Algorithm 1 Preference Elicitation for GRUMs

```
**Heuristic:** $G^*_R(\hat{\Theta}, h)$.
Randomly choose an initial set of data $D^1$.
**for** $t = 1$ to $T$ **do**
  **1:** Compute $\Theta^t = \text{MAP}(D^t)$.
  **2:** Compute the precision matrix $R^t$ of $\Pr(\Theta|D^t)$ at $\Theta^t$.
  **3:** Compute $h^t \in \mathscr{H}$ that maximizes $G^*_{R^t}(\Theta^t, h^t)$.
  **4:** Query an agent whose attributes are $h^t$. Let $\pi^t$ denote her preferences. $D^{t+1} \leftarrow D^t \cup \{\pi^t\}$, $\mathscr{H} \leftarrow \mathscr{H} \setminus \{h^t\}$.
**end for**
```

$$J_{D,h}(\hat{\Theta}) = \sum_{\pi \in D} (X_\pi \times (X_\pi)^T|_{\Theta=\hat{\Theta}}).$$

Equivalently, if $\log \Pr(\pi|\Theta, h)$ is twice differentiable w.r.t. Θ for each ranking π, then we have:

$$J_{D,h}(\hat{\Theta}) = -\sum_{\pi \in D} (\nabla^2_\Theta \log \Pr(\pi|\Theta,h)|_{\Theta=\hat{\Theta}}).$$

In Section 4 we propose an MC-EM algorithm to compute $\text{MAP}(D^t)$ in Step 1. In Section 4.3 we study how

[6] The elicitation scheme can be extended to other types of elicitation questions, for instance, pairwise comparisons and "top-k".

[7] See e.g. page 224 [2] for justification of this approximation.

Name	Quality function	Heuristics $G_R^*(\hat{\Theta}, h)$
D-optimality	Gain in Shannon information	$det(R + I_h(\hat{\Theta}))$
E-optimality	Minimum eigenvalue of the information matrix	$\lambda_{min}\{R + I_h(\hat{\Theta})\}$
Proposed criterion for social choice	Minimum inverse of pairwise coefficient of variation	Equation (5)
Proposed criterion for personalized choice	Minimum inverse of pairwise coefficient of variation	Equation (6)

Table 1: Different criteria for experimental design.

to compute the observed Fisher information matrix $R^t = J_{D^t}(\Theta^t)$, and use it for elicitation as well as accelerating MC-EC algorithm. Computation of the Fisher information matrix $I_h(\hat{\Theta})$ used in Step 3 will also be discussed in Section 4.3.

The choice of G_R^* is crucial for the performance of the elicitation algorithm. The two first criteria summarized in Table 1 are generic criteria for making the posterior as certain as possible, which may not work well for eliciting the aggregated ranking or individual rankings. In Section 6 we report experimental results comparing performance of different G_R^* in Table 1 and the new criteria we propose for both social choice and individual ranking.

3.1 A New Elicitation Criterion for Social Choice

The social choice ranking is the ranking over the components of $\vec{\delta}$. Therefore, if the objective is to elicit preferences for the aggregated ranking, it makes sense to make each pairwise comparison as certain as possible. Following the idea in t-test, we propose to use $\frac{|\text{mean}(\delta_{j_1} - \delta_{j_2})|}{\text{std}(\delta_{j_1} - \delta_{j_2})}$ (which is the inverse of *coefficient of variation*) to evaluate the certainty in pairwise comparison between c_{j_1} and c_{j_2}. The larger the value is, the more certain we are about the comparison between c_{j_1} and c_{j_2}. Therefore, we propose to use the following quality function G distributions over Θ. We recall that $\Theta = (\vec{\delta}, B)$.

$$G(\Pr) = \min_{j_1 \neq j_2} \frac{|\text{mean}(\delta_{j_1} - \delta_{j_2})|}{\text{std}(\delta_{j_1} - \delta_{j_2})}.$$

In words, G is the minimum inverse of the coefficient of variation across all pairwise comparisons. The corresponding G_R^* is thus the following.

$$G_R^*(\Theta, h) = \min_{j_1 \neq j_2} \frac{|\text{mean}(\delta_{j_1} - \delta_{j_2})|}{\sqrt{\text{Var}(\delta_{j_1}) + \text{Var}(\delta_{j_2}) + 2\text{cov}(\delta_{j_1}, \delta_{j_2})}}, \tag{5}$$

Where $|\text{mean}(\delta_{j_1} - \delta_{j_2})|$ can be computed from Θ and $\sqrt{\text{Var}(\delta_{j_1}) + \text{Var}(\delta_{j_2}) + 2\text{cov}(\delta_{j_1}, \delta_{j_2})}$ can be computed from $R + I_h(\Theta)$.

3.2 Generalization to Personalized Choice

Following the idea in the new criterion proposed in the last subsection for social choice, for any agent with attributes $\vec{x}$, we can define a similar quality function $G_{\vec{x}}(\Pr)$. This makes the ranking of the alternatives w.r.t. the deterministic parts of the perceived utilities[8] as certain as possible, as follows. For any $j \leq m$, let $\mu_j = \delta_j + \vec{x}B(\vec{z}_j)^T$. We note that μ_j is a linear combination of the parameters in Θ.

$$G_{\vec{x}}(\Pr) = \min_{j_1 \neq j_2} \frac{|\text{mean}(\mu_{j_1} - \mu_{j_2})|}{\text{std}(\mu_{j_1} - \mu_{j_2})} \tag{6}$$

$G_{\vec{x}}^*(\Theta, h)$ can be defined in a similar way. However, usually we want to predict the rankings for a population of agents, for which only a distribution over agent attributes is known. Mathematically, let Δ denote a probability distribution over $\mathbb{R}^L$. We can extend the criterion for personalized choice w.r.t. Δ as follows.

$$G_\Delta(\Pr) = \int_{\vec{x} \in \mathbb{R}^T} G_{\vec{x}}(\Pr) \cdot \Delta(\vec{x})\, d\vec{x}.$$

G_Δ is usually hard to compute since it involves integrating $G_{\vec{x}}$ over all $\vec{x}$ in support of Δ, which is often not analytically or computationally tractable. In the experiments, we will use the criterion defined in (5) for personalized ranking and surprisingly it works well.

4 An MC-EM Inference Algorithm

In this section, we extend MC-EM algorithm for RUMs proposed by Azari et al. [1] to GRUMs. We focus on GRUMs where the conditional probability $\Pr(\cdot|\vec{x}_i, \vec{z}_j, \delta_j, B)$ belongs to the *exponential family*, which takes the following form: $\Pr(U = u|\vec{x}_i, \vec{z}_j, \delta_j, B) = e^{\eta_{ij} \cdot T(u) - A(\eta_{ij}) + H(u)}$, where η_{ij} is the vector of *natural parameters*, which is a function of $\vec{x}_i, \vec{z}_j, \Theta$. A is a function of η_{ij} and T and H are functions of u.

Let $U = (\vec{u_1}, \ldots, \vec{u_n})$ denote the latent space, where $\vec{u_i} = (u_{i1}, \ldots, u_{im})$ represent agent i's perceived utilities for the alternatives. The general framework of the proposed EM algorithm is illustrated in Algorithm 2. The algorithm has multiple iterations, and in each iteration there is an E-step and a general M-step. Therefore, the algorithm is a *general EM (GEM)* algorithm. We recall that $\Theta = (\vec{\delta}, B)$ represents the parameters.

The algorithm is performed for a fixed number of iterations or until no Θ^{t+1} in the M-step can be found. However, the

[8]That is, the intrinsic utility plus personalized utility.

Algorithm 2 Framework of the EM algorithm

```
In each iteration.
E-Step : $Q(\Theta, \Theta^t)$
    $= E_{\vec{U}} \left\{ \log \prod_{i=1}^{n} \Pr(\vec{u}_i, \pi^i | \Theta) + \log(\Pr(\Theta)) | D, \Theta^t \right\}$   (7)
M-step : compute $\Theta^{t+1}$ s.t. $Q(\Theta^{t+1}, \Theta^t) > Q(\Theta^t, \Theta^t)$
```

E-step cannot be done analytically in general, and we will use a Monte Carlo approximation for the E-step.

4.1 Monte Carlo E-Step: Gibbs Sampling

Our E-step is similar to the E-step in [1] with a modification that considers the prior. We recall that $\Pr(\cdot|\vec{x}_i, \vec{z}_j, \delta_j, B)$ belongs to the exponential family. We have the following calculation for iteration t, where $\mu_{ij} = \delta_{ij} + \vec{x}_i B(\vec{z}_j)^T$ for any given $\Theta = (\vec{\delta}, B)$, and $\mu_{ij}^t = \delta_{ij}^t + \vec{x}_j B^t(\vec{z}_i)^T$.

$$Q(\Theta, \Theta^t) = E_{\vec{U}} \{ \log \prod_{i=1}^{n} \Pr(\vec{U}_i, \pi^i \mid \Theta) + \log \Pr(\Theta) \mid D, \Theta^t \}$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{m} E_{u_{ij}} \{ \log \Pr(u_{ij}|\Theta) \mid \pi^i, \Theta^t \}$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{m} \eta_{ij} S_{ij}^t - A(\eta_{ij}) + W,$$

$$\text{where} \quad S_{ij}^t = E_{u_{ij} \sim \Pr(u_{ij}|\eta_{ij}^t)} \{ u_{ij} | \pi^i \}. \tag{8}$$

We use a Monte Carlo approximation similar to that used in [1], which involves sampling U from the distribution $\Pr(U \mid D, \Theta^t)$ using a Gibbs sampler, and then approximate S_{ij}^{t+1} by $\frac{1}{N} \sum_{k=1}^{N} u_{ij}^k$. Each step of the Gibbs sampler is sampling from a truncated exponential distribution, illustrated in Figure 2 in [1].

4.2 General M-Step

After we compute S_{ij}^{t+1}'s, the M-step aims at improving $Q(\Theta, \Theta^t)$:

$$Q(\Theta, \Theta^t) = \sum_{j=1}^{m} \sum_{i=1}^{n} \log \Pr{}_j(u_{ij} = S_{ij}^{t+1} \mid \Theta) + \log(\Pr(\Theta))$$

We use steps of Newton's method to improve $Q(\Theta, \Theta^t)$ in the M-step (we can use as many steps at each iteration to ensure the convergence for each M-step).

$$\Theta^{t+1} = \Theta^t - (\nabla_\Theta^2 Q(\Theta, \Theta^t)|_{\Theta^t})^{-1} \nabla_\Theta Q(\Theta, \Theta^t)|_{\Theta^t} \tag{9}$$

$\nabla_\Theta^2 Q(\Theta, \Theta^t)$ and $\nabla_\Theta Q(\Theta, \Theta^t)$ can be computed immediately from S_{ij}^t as follows.

$$\nabla_\Theta^2 Q(\Theta, \Theta^t) = \sum_{i=1}^{n} \sum_{j=1}^{m} \nabla_\Theta^2 \eta_{ij} S_{ij}^t - \nabla_\Theta^2 A(\eta_{ij})$$
$$\nabla_\Theta Q(\Theta, \Theta^t) = \sum_{i=1}^{n} \sum_{j=1}^{m} \nabla_\Theta \eta_{ij} S_{ij}^t - \nabla_\Theta A(\eta_{ij})$$

4.3 Computing Observed Fisher information

Computation of the observed Fisher information will not only be used in Step 2 of the new elicitation scheme Algorithm 1, but also will accelerate the GEM algorithm [16]. Fisher information can be computed by the following method proposed by Louis [16]. From the independence of agents we have: $J_D(\hat{\Theta}) = \sum_i J_{\pi^i}(\hat{\Theta})$, where,

$$J_{\pi^i}(\Theta) = E_{U_i} \{ -\nabla_\Theta^2 \log P(\pi^i, U_i|\Theta) | \Theta, \pi^i \}$$
$$- E_{U_i} \{ \nabla_\Theta \log P(\pi^i, U_i|\Theta) \nabla_\Theta \log P(\pi^i, U_i|\Theta)^T | \Theta, \pi^i \}$$

$J_{\pi^i}(\hat{\Theta})$ is computed using the samples (u_{ij}'s) generated in MC step in every iteration of EM algorithm as follows.

$$\nabla_\Theta^2 \log P(\pi^i, U_i|\Theta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \nabla_\Theta^2 \eta_{ij} U_{ij} - \nabla_\Theta^2 A(\eta_{ij})$$
$$\nabla_\Theta \log P(\pi^i, U_i|\Theta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \nabla_\Theta \eta_{ij} U_{ij} - \nabla_\Theta A(\eta_{ij})$$

The Fisher information matrix $I_h(\hat{\Theta})$ used in Step 3 of Algorithm 1 can be approximated by $\lim_{n \to \infty} \frac{J_{D_n}(\hat{\Theta})}{n}$, where D_n is the dataset of n rankings randomly generated according to $\Pr(\pi|\hat{\Theta})$. Therefore, we can use the techniques developed in this subsection to approximately compute $I_h(\hat{\Theta})$.

4.4 MC-EM Algorithm in Detail

The details of the proposed EM algorithm (with fixed number of iterations) are illustrated in Algorithm 3.

Algorithm 3 MAP for GRUM

```
Input: $D = (\pi^1, \ldots, \pi^n)$, $\Theta^{start}$, $T \in \mathbb{N}$
Let $\Theta^0 = \Theta^{start}$
for $t = 1$ to $T$ do
    for every $\pi^i \in D$ do
        Compute $S_{ij}^{t+1}$ and $J(\Theta^{t+1})$ according to (8) for all $j \leq m$.
    end for
    Compute $\Theta^{t+1}$ according to (9).
end for
```

5 Global Optimality for Posterior Distribution

In this section, we generalize theorems on global optimality of likelihood for RUMs proved in [1] to GRUMs. All proofs are omitted due to the space constraint. The EM algorithm tends to find local optimal of the posterior distribution, hence, proving global optimality of MAP helps to avoid issues due to EM. First, we present concavity of the posterior distribution in GRUMs.

Theorem 1 *For the location family, if for every $j \leq m$ the joint probability density function for $\vec{\epsilon_i}$ and the prior $\Pr(\Theta)$ are log-concave, then $\Pr(\Theta|D)$ is concave up to a known transformation.*

For P-L, Ford, Jr. [12] proposed the following necessary and sufficient condition for the set of global maxima solutions to be bounded (more precisely, unique) when $\sum_{j=1}^{m} e^{\Theta_j} = 1$. The conditions are generalized to the case of RUMs in [1]. We prove that this condition is also necessary and sufficient for global maxima solutions of the likelihood function of GRUMS to be bounded.

Condition 1 *Given the data D, in every partition of the alternatives $\mathcal{C}$ into two nonempty subsets $\mathcal{C}_1 \cup \mathcal{C}_2$, there exists $c_1 \in \mathcal{C}_1$ and $c_2 \in \mathcal{C}_2$ such that there is at least one ranking in D where $c_1 \succ c_2$.*

Theorem 2 *Suppose we fix $\mu_{11} = 0$. Then, the set S_D of global maxima solutions to $\Pr(\Theta|D)$ is bounded in Θ if and only if the data D satisfies Condition 1 and the linear model describing μ in terms of Θ is identifiable.*

6 Experimental Results

In this section, we report experimental results on synthetic data and a Sushi dataset from Kamishima [14] for three types of tests described below.

6.1 Social Choice and Synthetic Data

We first show the consistency of the model for social choice. We generate random data sets with $\delta_j \sim \text{Normal}(1,1), B_{ij} \sim \text{Normal}(0,1), X_i \sim \text{Normal}(0,1), Z_i \sim \text{Normal}(0,1)$, and then generate random utilities with the random noise ϵ_{ij} generated with mean zero and variance of 1. The results in Figure 2 are generated by varying the number of agents for which we have preference information. For each number of agents, we estimate the parameter set Θ, and evaluate the Kendall correlation between estimated and true ranks with respect to δ_j's. These results illustrate the improvement in estimated social choice order as the number of agents in the population increases.

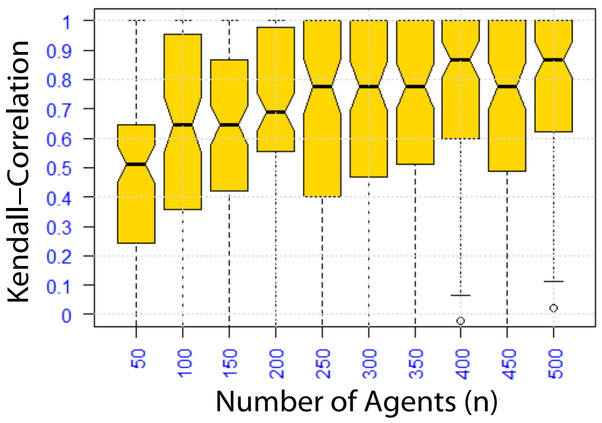


Figure 2: Asymptotic behavior for synthetic data and social choice. The y-axis is the average Kendal correlation between the estimated social choice and the ground truth order.

In studying elicitation for social choice, we test the performance of the elicitation schemes shown in Table 1, i.e. D-optimality, E-optimality, and the proposed criterion in (5), and compare the results to random elicitation. We adopt the following two synthetic datasets:

Dataset 1: $(B_{ij} \sim N(0,1), X_i \sim N(0,1), Z_i \sim N(0,1)), \delta_j \sim 0.1 * N(1,1)$ and the error term $\epsilon_{ij} \sim N(0,1)$.

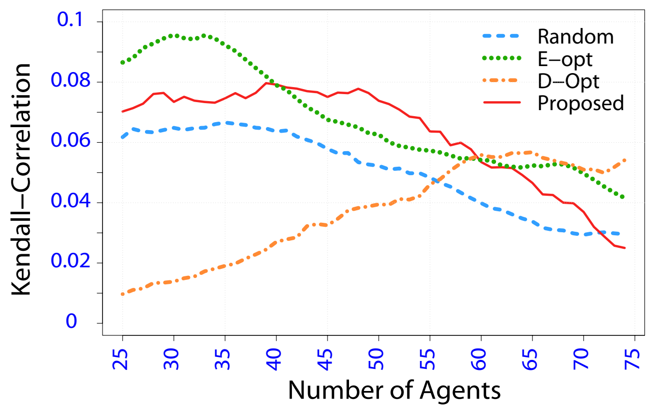
Dataset 2: The same as Dataset 1, except that the $\delta_j \sim N(1,1)$ and the error term $\epsilon_{ij} \sim N(0,1/4)$.

Compared to the GRUM in Dataset 1, the model adopted in Dataset 2 has a heavier social component and less noise. For each dataset we generate 100 agents' preferences, and use the three criteria shown in Table 1 to elicit $n \in [1,100]$ rankings. For each n, we apply Algorithm 3 and compare the ranking over the learned δ_j's with the ground truth social choice ranking.

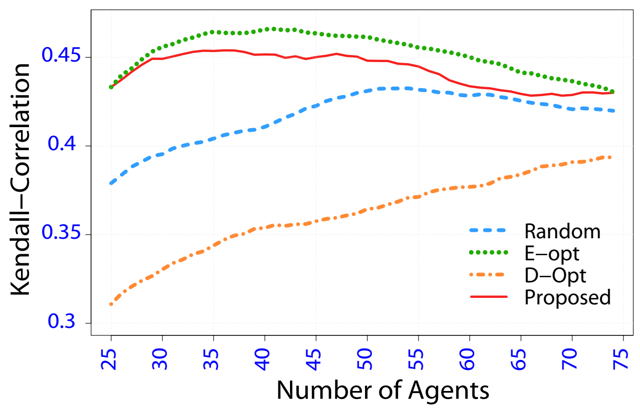
The results are shown in Figure 3 (graphs are smoothed with a moving window with length 25), where the x-axis is the number of agents whose preferences are elicited, and the y-axis is the Kendall correlation between the learned ranking and the ground truth ranking. We make the following observations.

• In Dataset 1 where the social component is small, it is not clear which criteria is better, as shown in Figure 3(a), and there are no statistically significant results.

• In Dataset 2 where the social component is large, E-optimality generally works better than the proposed method, while both work better than random, which works surprisingly better than D-optimality, as shown in Figure 3(b). However, only a few of these observations are statistically significant with 90% confidence, for example, considering the interval of $[34,44]$ agents, E-optimality and the proposed method outperforms Random but the comparison between the other methods is not significant at 90%.

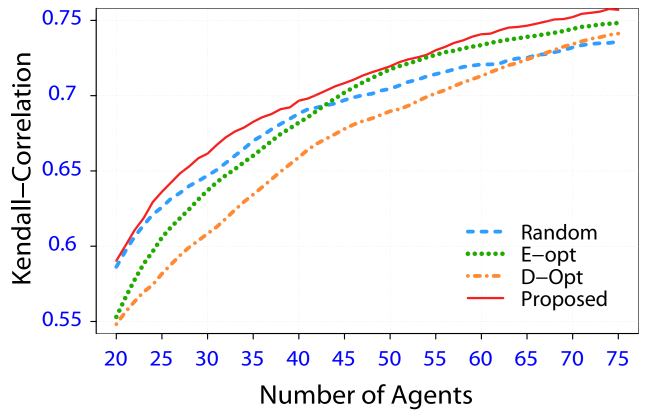


(a) Social choice: Dataset 1.

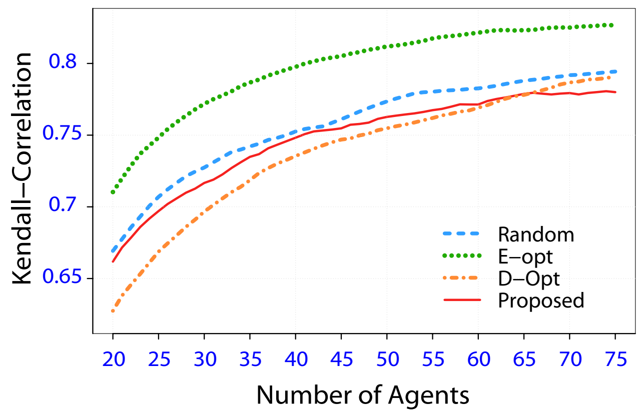


(b) Social choice: Dataset 2.

Figure 3: Comparison of elicitation criteria described in Table 1 for synthetic data and social choice.

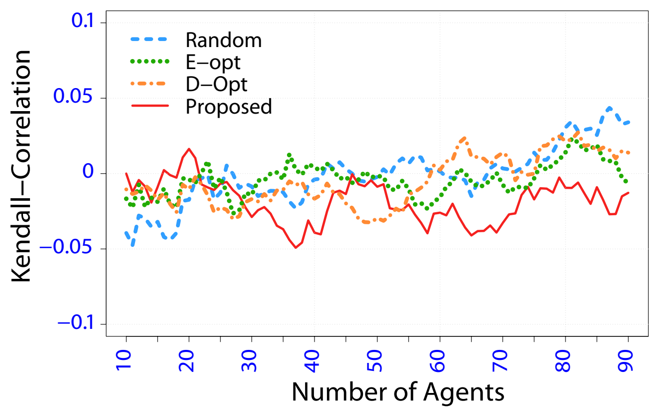


(a) Personalized choice: Dataset 1.

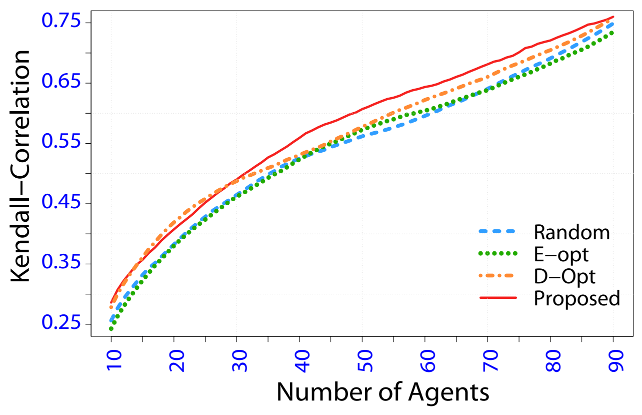


(b) Personalized choice: Dataset 2.

Figure 4: Comparison of elicitation criteria described in Table 1 for synthetic data for personalized choice.



(a) Social choice: Sushi dataset.



(b) Personalized choice: Sushi dataset.

Figure 5: Comparison of elicitation criteria described in Table 1 for the Sushi dataset [14].

6.2 Personalized Choice and Synthetic Data

For personalized choice we first show the consistency results in Figure 6, where the bottom box-plot shows the Kendall correlation between noisy data (i.e., an individual agent's random utility and thus preference order) and the true preference order for each agent, and the top box-plot shows Kendall correlation between estimated agent preference orders and true preference orders, as obtained through the model.

Turning to preference elicitation, we compare the schemes in Table 1 with the random method for the same two datasets as were adopted for social choice. The results are shown in Figure 4(graphs are smoothed with a moving window with length 20). For each group of 100 agents, and for any $n \in [1, 100]$ and each elicitation scheme, we compute the MAP of Θ, and use it to compute the Kendall correlation between the true preferences and the predicted preference for all 100 agents in this group. We make the following observations:

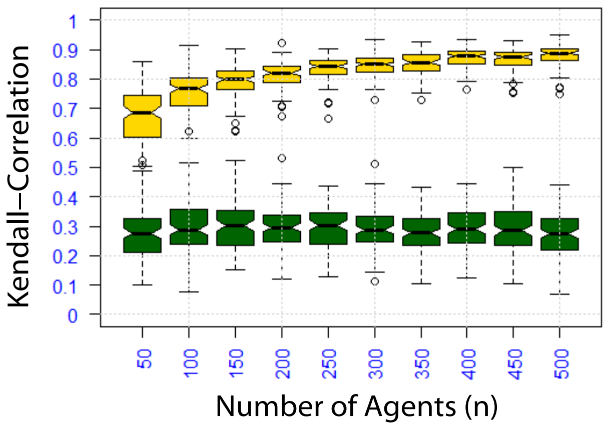


Figure 6: Asymptotic behavior for synthetic data and personalized choice. The y-axis is the average Kendall correlation between an estimated preference order and ground truth preference order for an agent. The top box-plot shows the result of inference, the bottom box-plot the correlation from raw data.

• In Dataset 1, where the social component is small, when the number of agents used in elicitation is not too large (< 50), the proposed method works better than E-optimality, which is itself comparable to random. Both methods are better than D-optimality. See Figure 4(a). Some of these observations are statistically significant, for example, when $n = [24, 25]$, E-optimality works better than D-optimality with 90% significance, E-optimality works better than random with 75% significance, the proposed method works better than E-optimality with 75% significance, and the proposed method works better than D-optimality with 75% significance.

• In Dataset 2, where the social component is large, E-optimality generally works better than the proposed method, both work better than random, and random is more effective than D-optimality, as shown in Figure 4(b). However, only a few of these observations are statistically significant with 90% confidence interval, for example E-optimality outperforms D-optimality when the number of agents is in the interval $[29, 42]$.

6.3 Sushi Data

In synthetic experiments, we have access to the ground truth. However, in the real world data (Sushi data) there are no data available as ground truth. In this experiment, we estimated parameters Θ using preferences from 1000 agents, randomly chosen from the 5000 agents in the dataset. And adopt those parameters as the ground truth for the experimental study. The categorical features are discarded from the data set.[9]

The results are shown in Figure 5 (graphs are smoothed with a moving window with length 10), where (a) shows comparisons for social choice (where we rank δ's), and (b) shows comparisons for personalized choice. We make the following observations:

• For social choice (a), none of the criteria work well (and note that the Kendall correlations are low for all criteria). We feel that this is reasonable since preferences over sushi is likely high personalized with a small social component to preferences.

• For personalized choice (b), we observe that the proposed method is generally the most effective, while the performance of E-optimality and D-optimality is very close to random. None of these results are statistically significant with 90% confidence.

7 Conclusion and Future Work

We have proposed a method for preference elicitation based on ordinal rank data, adopting the framework of Bayesian experimental design. This includes two new criteria for social and personalized case. The proposed criterion for social choice can significantly improve the precision of estimation, relative to random and some of the classical elicitation criteria. This work can also be seen as preference elicitation for learning to rank. In the future, we can adopt the methodology in other preference elicitation applications; for example recommendation systems, product prediction and so forth.

Acknowledgments

This work is supported in part by NSF Grant No. CCF-0915016. Lirong Xia is supported by NSF under Grant #1136996 to the Computing Research Association for the CIFellows Project. We thank Simon Lunagomez for providing valuable comments on the Bayesian experimental design aspect of this work. We also thank Edoardo Airoldi, Craig Boutilier, Jonathan Huang, Tyler Lu, Nicolaus Tideman, Paolo Viappiani, and anonymous UAI-13 reviewers, for helpful comments and suggestions, or help on the datasets.

[9]We focus on non categorical features in this work. The method can be extended to categorical features.

References

[1] Hossein Azari Soufiani, David C. Parkes, and Lirong Xia. Random utility theory for social choice. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pages 126–134, Lake Tahoe, NV, USA, 2012.

[2] James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. James O. Berger, 2nd edition, 1985.

[3] Steven Berry, James Levinsohn, and Ariel Pakes. Automobile prices in market equilibrium. *Econometrica*, 63(4):841–890, 1995.

[4] Steven Berry, James Levinsohn, and Ariel Pakes. Differentiated products demand systems from a combination of micro and macro data: The new car market. *Journal of Political Economy*, 112(1):68–105, 2004.

[5] Edwin Bonilla, Shengbo Guo, and Scott Sanner. Gaussian process preference elicitation. In *Advances in Neural Information Processing Systems 23*, pages 262–270. 2010.

[6] Craig Boutilier. On the foundations of expected expected utility. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 285–290, Acapulco, Mexico, 2003.

[7] Craig Boutilier. Computational Decision Support: Regret-based Models for Optimization and Preference Elicitation. In P. H. Crowley and T. R. Zentall, editors, *Comparative Decision Making: Analysis and Support Across Disciplines and Applications*. Oxford University Press, 2013.

[8] Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[9] Urszula Chajewska, Daphne Koller, and Ron Parr. Making rational decisions using adaptive utility elicitation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 363–369, Austin, TX, USA, 2000.

[10] Kathryn Chaloner and Isabella Verdinelli. Bayesian Experimental Design: A Review. *Statistical Science*, 10(3):273—304, 1995.

[11] Vincent Conitzer and Tuomas Sandholm. Vote elicitation: Complexity and strategy-proofness. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 392–397, Edmonton, AB, Canada, 2002.

[12] Lester R. Ford, Jr. Solution of a ranking problem from binary comparisons. *The American Mathematical Monthly*, 64(8):28–33, 1957.

[13] Neil Houlsby, Jose Miguel Hernandez-Lobato, Ferenc Huszar, and Zoubin Ghahramani. Collaborative gaussian processes for preference learning. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2105–2113. Lake Tahoe, NV, USA, 2012.

[14] Toshihiro Kamishima. Nantonac collaborative filtering: Recommendation based on order responses. In *Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 583–588, Washington, DC, USA, 2003.

[15] Jérôme Lang and Lirong Xia. Sequential composition of voting rules in multi-issue domains. *Mathematical Social Sciences*, 57(3):304–324, 2009.

[16] Thomas A. Louis. Finding the observed information matrix when using the EM algorithm. *Journal of the Royal Statistical Society Series B (Statistical Methodology)*, 44:226–233, 1982.

[17] Tyler Lu and Craig Boutilier. Robust approximation and incremental elicitation in voting protocols. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, pages 287–293, Barcelona, Catalonia, Spain, 2011.

[18] Robert Duncan Luce. *Individual Choice Behavior: A Theoretical Analysis*. Wiley, 1959.

[19] Thomas Pfeiffer, Xi Alice Gao, Andrew Mao, Yiling Chen, and David G. Rand. Adaptive Polling and Information Aggregation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 122–128, Toronto, Canada, 2012.

[20] Robin L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 24(2):193–202, 1975.

[21] Tuomas Sandholm and Craig Boutilier. Preference elicitation in combinatorial auctions. In Peter Cramton, Yoav Shoham, and Richard Steinberg, editors, *Combinatorial Auctions*, chapter 10, pages 233–263. MIT Press, 2006.

[22] Louis Leon Thurstone. A law of comparative judgement. *Psychological Review*, 34(4):273–286, 1927.

[23] Joan Walker and Moshe Ben-Akiva. Generalized random utility model. *Mathematical Social Sciences*, 43(3):303–343, 2002.

Calculation of Entailed Rank Constraints in Partially Non-Linear and Cyclic Models

Peter Spirtes
Department of Philosophy
Carnegie Mellon University
ps7z@andrew.cmu.edu

Abstract

The Trek Separation Theorem (Sullivant et al. 2010) states necessary and sufficient conditions for a linear directed acyclic graphical model to entail for all possible values of its linear coefficients that the rank of various sub-matrices of the covariance matrix is less than or equal to *n*, for any given *n*. In this paper, I extend the Trek Separation Theorem in two ways: I prove that the same necessary and sufficient conditions apply even when the generating model is partially non-linear and contains some cycles. This justifies application of constraint-based causal search algorithms to data generated by a wider class of causal models that may contain non-linear and cyclic relations among the latent variables.

1 INTRODUCTION

In many cases, scientists are interested in inferring causal relations between variables that cannot be directly measured (e.g. intelligence, anxiety, or impulsiveness) by administering test surveys with measured "indicators" that indirectly measure the unmeasured or "latent" variables. In other cases, scientists are interested in estimating the values of the latent variables from the measured indicators. The variances of the estimates of the latent variables of interest can be reduced in various ways by employing multiple indicators for each latent variable. A model in which each latent variable of interest is measured by multiple indicators (which may also be caused by other latents of interest as well as the error variable) is called a multiple indicator model. Multiple indictor models are quite common in many disciplines such as educational research, psychology, political science, etc. (Bartholomew et al., 2002).

Two major problems are how to use the values of the measured indicator variables to make reliable inferences about the causal relationships between the latent variables of interest, and to predict the values of the latent variables from the values of the measured indicators. A number of complications make both of these tasks very difficult:

- Associations among indicators are often confounded by additional unknown latent common causes;
- One indicator may directly affect other indicators (e.g. "anchoring effects");
- There are often a plethora of alternative causal models that are consistent with the data and with the prior knowledge of domain experts, often far too many models to test individually;
- There may be non-linear dependencies among latent variables;
- There may be feedback relationships among latent variables.

The most common algorithms for using measured indicators to find causal relations among latent variables or to infer the values of the latent variables use some version of factor analysis. However, given the models with the features cited above, factor analytic algorithms, as well as the FindHidden algorithm of Elidan (2001) have been shown to perform poorly (Silva et al. 2006).

One class of model search algorithms that have had some success dealing with some of the complications listed above is constraint-based search. A constraint-based search attempts to find the set of models that most closely match the measured constraints on a probability distribution that are entailed for all values of the free parameters (e.g. conditional independence constraints that are entailed by d-separation) with constraints that are judged to hold in the population (as determined by a statistical test).

Although multiple indicator models rarely entail any conditional independence constraints among just the measured indicators, multiple indicator models often entail constraints on the rank of sub-matrices of the covariance matrix among the measured indicators (e.g. vanishing tetrad differences explained below), and there are searches based on these rank constraints that have desirable properties (Silva et al. 2006).

Multiple indicator models are special cases of structural equation models, and the form of the equations can be represented by a directed graph (Pearl 2000, Spirtes et al. 2001). Under the assumption of linearity, the graphical structure representing the multiple indicator model can *linearly entail* constraints on the covariance matrix of the variables, that is, constraints that hold for all values of the free parameters (the linear coefficients associated with the edges, and the variances of the error terms). For example, a multiple indicator model represented by a graph with a single latent variable L that is the parent of measured indicators X, Y, Z, and W and contains no other edges, entails the vanishing tetrad difference (e.g. $\rho(X,Y)\rho(Z,W) - \rho(X,Z)\rho(Y,W) = 0$) for all values of the linear coefficients, which is equivalent to a constraint that the rank of a submatrix of the covariance matrix is less than or equal to 1.

The Trek Separation Theorem (Sullivant et al. 2010) states necessary and sufficient conditions for a directed acyclic graph to linearly entail that the rank of various sub-matrices of the covariance matrix among the measured variables are less than or equal to n, for any n.

The Trek Separation Theorem is one way to justify the correctness of the BuildPureClusters algorithm (Silva et al. 2006), that searches for the set of multiple indicator models that most closely match the set of vanishing tetrad differences judged to hold in the population by application of statistical tests to the sample data. BuildPureClusters is a pointwise consistent algorithm that, depending upon the input data, either outputs "Can't tell" or an equivalence class of graphs that linearly entail the same set of vanishing tetrad differences and zero partial correlation constraints. The algorithm has been successfully applied to a number of data sets (Silva et al. 2006, Jackson & Scheines 2005)

However, there are a number of significant limitations on usefulness of the Trek Separation Theorem (and hence on the BuildPureClusters algorithm):

- The Trek Separation Theorem does not apply to cyclic graphs (as in feedback models);
- The Trek Separation Theorem does not apply if any of the causal relations between the variables are non-linear.

In this paper, I prove an extension of the trek separation theorem which gives necessary and sufficient conditions for a directed graph (cyclic or acyclic) that has some functions relating variables to other variables that are non-linear, and in which there may be some feedback (represented by cyclic graphs) to entail that the rank of various sub-matrices of the covariance matrix are less than or equal to n, for any n. This theorem has at least two uses for causal discovery: it serves as the basis for proving that existing algorithms for the linear case can be reliably applied to partially non-linear or cyclic models (described in section 4), and it could be used in the development of new algorithms for causal inference among models in which measured indicators have multiple latent parents but have non-linear or cyclic relations among the latent parents.

In section 2, I describe multiple indicator models and the Trek Separation Theorem in more detail. In section 3, I state an extension of the trek separation theorem that applies to graphs that may have cyclic and non-linear relationships among some variables. In section 4, I discuss the issue of the extent to which it is to be expected that rank constraints on the covariance matrix might hold, or approximately hold, in the population even if they are not entailed by the model to hold for all values of the free parameters of the model. In section 5, I describe open research questions. The Appendix contains the proofs.

2 STRUCTURAL EQUATION MODELS

In what follows, random variables are in italics, and sets of random variables are in boldface..

In a structural equation model (SEM) the random variables are divided into two disjoint sets, the substantive variables (typically the variables of interest) and the error variables (summarizing all other variables that have a causal influence on the substantive variables) (Bollen, 1989). Corresponding to each substantive random variable V is a unique error term ε_V. A *fixed parameter SEM S* has two parts $<\boldsymbol{\phi}, \boldsymbol{\theta}>$, where $\boldsymbol{\phi}$ is a set of equations in which each substantive random variable V is written as a function of other substantive random variables and a unique error variable, together with $\boldsymbol{\theta}$, the joint distributions over the error variables. An example of a linear SEM is the case where $\boldsymbol{\phi}$ contains the pair of linear equations $X = 3L + \varepsilon_X$, and $L = \varepsilon_L$, and $\boldsymbol{\theta}$ is a standardized Gaussian distribution over ε_X and ε_L and ε_X and ε_L are independent. Together $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ determine a joint distribution over the substantive variables in S, which will be referred to as the *distribution entailed by S*.

A *free parameter* linear SEM model replaces some of the real numbers in the equations in ϕ with real-valued variables and a set of possible values for those variables, e.g. $X = a_{X,L}\, L + \varepsilon_X$, where $a_{X,L}$ can take on any real value. In addition, a free parameter SEM can replace the particular distribution over ε_X and ε_L with a parametric family of distributions, e.g. the bi-variate Gaussian distributions with zero covariance. The free parameter SEM also has two parts $<\boldsymbol{\Phi}, \boldsymbol{\Theta}>$, where $\boldsymbol{\Phi}$ contains the set of equations with free parameters and the set of values the free parameters are allowed to take, and $\boldsymbol{\Theta}$ is a family of distributions over the error variables.

In general, I will assume that there is a finite set of free parameters, and all allowed values of the free parameters lead to fixed parameter SEMs that have a reduced form (i.e. each substantive variable X can be expressed as a function of the error variables of X and the error variables of its ancestors), all variances and partial variances among the substantive variables are finite and positive, and there are no deterministic relations among the measured variables.

The path diagram of a SEM with jointly independent errors is a directed graph, written with the conventions that it contains an edge $B \rightarrow A$ if and only if B is a non-trivial argument of the equation for A. The error variables are not included in the path diagram. A fixed-parameter acyclic structural equation model (without double-headed arrows) is an instance of a Bayesian Network $<G, P(\mathbf{V})>$, where the path diagram is G, and $P(\mathbf{V})$ is the joint distribution over the variables in G entailed by the set of equations and the joint distribution over the error variables (Pearl, 2000; Spirtes et al. 2001). It has been shown that when a directed cyclic graph is used to represent non-linear structural equations, then d-separation between **A** and **B** conditional on **C** does not entail the corresponding conditional independence. Even in non-linear cyclic structural equation models, if **A** and **B** are d-separated conditional on the empty set, then **A** and **B** are entailed to be independent (Spirtes, 1995), and that is the only feature of cyclic graphs that the proofs below depend upon.

A polynomial equation Q on the entries of a covariance (or correlation) matrix **C** *holds* when **C** is a solution to Q. A polynomial Q is *entailed by a free parameter* SEM when all values of the free parameters entail covariance matrices that are solutions to Q.

For example, a *vanishing tetrad difference* among $\{X,W\}$ and $\{Y,Z\}$, which holds if $\rho(X,Y)\rho(Z,W) - \rho(X,Z)\rho(Y,W) = 0$, is entailed by a free parameter SEM S in which X, Y, Z, and W are all children of just one latent variable L since any value of the free parameters in S entails a covariance matrix that is a solution to $\rho(X,Y)\rho(Z,W) - \rho(X,Z)\rho(Y,W) = 0$.

The following definitions are illustrated in Figure 1. A *trek* in G from I to J is an ordered pair of directed paths (P_1; P_2) where P_1 has sink I, P_2 has sink J, and both P_1 and P_2 have the same source k (e.g. $(<L_1,X_1>;<L_1,X_2>)$. The common source k is called the *top* of the trek, denoted *top*(P_1; P_2) (e.g. top($<L_1,X_1>;<L_1,X_2>$) is L_1). Note that one or both of P_1 and P_2 may consist of a single vertex, i.e., a path with no edges. A trek (P_1; P_2) is *simple* if the only common vertex between P_1 and P_2 is the common source *top*(P_1; P_2). Let **A**, **B**, be two disjoint subsets of vertices **V** in G. Let **T**(**A**,**B**) and **S**(**A**,**B**) denote the sets of all treks and all simple treks from a member of **A** to a member of **B**, respectively. For example, if **A** = $\{X_1\}$ and **B** = $\{X_2\}$, **S**(**A**,**B**) = $\{(<L_1,X_1>;<L_1,X_2>); (<L_2,X_1>;<L_2,X_2>)\}$.

For two sets of variables **A** and **B**, and a covariance or correlation matrix over a set of variables **V** containing **A** and **B**, let cov(**A**, **B**) be the sub-matrix of **Σ** that contains the rows in **A** and columns in **B**. For example, if **A** = $\{X_1, X_2, X_3\}$, and **B** = $\{X_4, X_5, X_{10}\}$, then

$$\mathrm{cov}(\mathbf{A},\mathbf{B}) = \begin{array}{c|ccc} & X_4 & X_5 & X_{10} \\ \hline X_1 & \rho(X_1,X_4) & \rho(X_1,X_5) & \rho(X_1,X_{10}) \\ X_2 & \rho(X_2,X_4) & \rho(X_2,X_5) & \rho(X_2,X_{10}) \\ X_3 & \rho(X_3,X_4) & \rho(X_3,X_5) & \rho(X_3,X_{10}) \end{array}$$

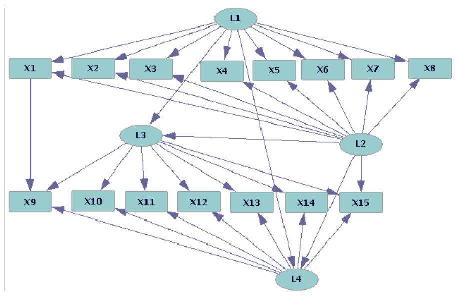


Figure 1: A Multiple Indicator Model

In the case where **A** and **B** both have size 3, if the rank of the matrix is less than or equal to 2, the determinant of cov(**A**,**B**) = 0. In that case the matrix is said to satisfy a *sextad constraint*. An example of a sextad constraint is

$$Det\left(\begin{bmatrix} \rho(X_1,X_4) & \rho(X_1,X_5) & \rho(X_1,X_{10}) \\ \rho(X_2,X_4) & \rho(X_2,X_5) & \rho(X_2,X_{10}) \\ \rho(X_3,X_4) & \rho(X_3,X_5) & \rho(X_3,X_{10}) \end{bmatrix}\right) = 0$$

Let **A**, **B**, $\mathbf{C_A}$, and $\mathbf{C_B}$ be four subsets of the set **V** of vertices in G, which need not be disjoint. The pair ($\mathbf{C_A}$;$\mathbf{C_B}$) *trek separates* (or *t-separates*) **A** from **B** if for every trek (P_1; P_2) from a vertex in **A** to a vertex in **B**, either P_1 contains a vertex in $\mathbf{C_A}$ or P_2 contains a vertex in $\mathbf{C_B}$; $\mathbf{C_A}$ and $\mathbf{C_B}$ are *choke sets* for **A** and **B**. For example, $(\{L_1\}; \{L_2\})$, $(\{L_1, L_2\}; \varnothing)$, and $(\varnothing; \{L_1,L_2\})$ all t-separate **A** from **B** in this example.

Theorem 1 (Trek Separation Theorem): For all directed acyclic graphs (path diagrams) G, the sub-matrix cov(**A**,**B**) has rank less than or equal to r for all covariance matrices of linear SEMs with path diagram G, if and only if there exist subsets $\mathbf{C_A}$, $\mathbf{C_B} \subset \mathbf{V}(G)$ with $\#\mathbf{C_A} + \#\mathbf{C_B} \leq r$ such that ($\mathbf{C_A}$; $\mathbf{C_B}$) t-separates **A** from **B** (where $\#\mathbf{C_A}$ is the number of variables in $\mathbf{C_A}$, and **V**(G) is the set of vertices in G). (Sullivant et al., 2010)

Since the rank of cov(**A**, **B**) is less than or equal to r, if $\mathbf{C_A} \cap \mathbf{C_B} = \varnothing$, $\#\mathbf{A} = \#\mathbf{B} = 3$, $\#\mathbf{C_A} + \#\mathbf{C_B} = 2$, and ($\mathbf{C_A}$; $\mathbf{C_B}$) t-separates **A** from **B**, then G entails a sextad constraint among the variables in **A** and **B**. For example, in Figure 1, $(\{L_1, L_2\};\{\ \})$ trek separates $\{X_1, X_2, X_3\}$ from $\{X_4, X_5, X_{10}\}$, and hence

$$rank\left(\begin{bmatrix} \rho(X_1,X_4) & \rho(X_1,X_5) & \rho(X_1,X_{10}) \\ \rho(X_2,X_4) & \rho(X_2,X_5) & \rho(X_2,X_{10}) \\ \rho(X_3,X_4) & \rho(X_3,X_5) & \rho(X_3,X_{10}) \end{bmatrix}\right) \leq \#\mathbf{C_A} + \#\mathbf{C_B} = 2$$

which in turn entails that the determinant of the matrix is zero for all values of the free parameters in a linear SEM.

3 AN EXTENSION OF THE TREK SEPARATION THEOREM

The Trek Separation Theorem can be extended by weakening the assumptions that the graph be linear everywhere and acyclic everywhere. The exact definition of *linear acyclicity* (or LA for short) *below* a *choke* set is somewhat complex (and is given below), but roughly a directed graphical model is *LA below sets* ($\mathbf{C_A}$; $\mathbf{C_B}$) for **A** and **B** respectively, if there are no directed cycles between $\mathbf{C_A}$ and **A** or $\mathbf{C_B}$ and **B**, and for every vertex V on any directed path P from $\mathbf{C_A}$ to **A**, V is a linear function of its parents along P plus an arbitrary function of the parents not along P (including the error variables); and similarly for $\mathbf{C_B}$ and **B**. For example in Figure 1, let the sets $\mathbf{C_A}$ and $\mathbf{C_B}$ for $\mathbf{A} = \{X_1, X_2, X_3\}$, and $\mathbf{B} = \{X_4, X_5, X_{10}\}$ be $\mathbf{C_A} = \{L_1, L_2\}$ and $\mathbf{C_B} = \varnothing$. Linear acyclicity below the sets $\mathbf{C_A}$, $\mathbf{C_B}$, for **A**, **B** requires that for $i = 1\ldots3$, $X_i = a_{i,1} L_1 + a_{i,2} L_2 + f_i(\varepsilon_i)$, where ε_i is the error term for X_i, and f_i is an arbitrary measurable function. (Since $\mathbf{C_B} = \varnothing$, linear acyclicity below the set $\mathbf{C_B}$ is trivially true). Note that there can be non-linear and/or cyclic relationships between any of the latent variables.

More formally, let $D(\mathbf{C_A},\mathbf{A},G)$ be the set of vertices on directed paths in G from $\mathbf{C_A}$ to **A** except for the members of $\mathbf{C_A}$ (but including members of $\mathbf{A}\backslash\mathbf{C_A}$). If S is a fixed-parameter SEM $<\phi,\theta>$ with path diagram G, *S is LA below the sets* $\mathbf{C_A}$, $\mathbf{C_B}$ *for* **A***,* **B** iff for each member of $\mathbf{W} = D(\mathbf{C_A},\mathbf{A},G) \cup D(\mathbf{C_B},\mathbf{B},G)$,

(i) $\mathbf{V_{ext}} = \mathbf{V} \cup \{\varepsilon_X : X \in \mathbf{W}\}$;

(ii) no member of **W** lies on a cycle;

(iii) G_{ext} is a directed graph over $\mathbf{V_{ext}}$ with sub-graph G, together with an edge from ε_X to X for each $X \in \mathbf{W}$,

(iv) for each $X \in \mathbf{D}(\mathbf{C_A},\mathbf{A},G_{ext})$,

$$X = \sum_{V \in \mathbf{Parents}(X,G_{ext}) \cap (D(\mathbf{C_A},\mathbf{A},G_{ext}) \cup \mathbf{C_A})} a_{X,V} V + f_X(\mathbf{Parents}(X,G_{ext}) \setminus (D(\mathbf{C_A},\mathbf{A},G_{ext}) \cup \mathbf{C_A})) \quad (1)$$

and for each $X \in \mathbf{D}(\mathbf{C_B},\mathbf{B},G_{ext})$,

$$X = \sum_{V \in \mathbf{Parents}(X,G_{ext}) \cap (D(\mathbf{C_B},\mathbf{B},G_{ext}) \cup \mathbf{C_B})} a_{X,V} V + g_X(\mathbf{Parents}(X,G_{ext}) \setminus (D(\mathbf{C_B},\mathbf{B},G_{ext}) \cup \mathbf{C_B})) \quad (2)$$

Note that $D(\mathbf{C_A},\mathbf{A},G) = D(\mathbf{C_A},\mathbf{A},G_{ext})$ for any **A** and $\mathbf{C_A}$ that do not contain an error variable.

Theorem 2 (Extended Trek Separation Theorem): Suppose G is a directed graph containing $\mathbf{C_A}$, **A**, $\mathbf{C_B}$, and **B**, and ($\mathbf{C_A}$;$\mathbf{C_B}$) t-separates **A** and **B** in G. Then for all covariance matrices entailed by a fixed parameter structural equation model S with path diagram G that is LA below the sets $\mathbf{C_A}$ and $\mathbf{C_B}$ for **A** and **B**, $rank(\text{cov}(\mathbf{A},\mathbf{B})) \leq \#\mathbf{C_A} + \#\mathbf{C_B}$.

The converse of Theorem 2 is basically guaranteed by the "only-if" clause of Theorem 1.

Theorem 3: For all directed graphs G, if there does not exist a pair of sets $\mathbf{C'_A}$, $\mathbf{C'_B}$, such that ($\mathbf{C'_A}$; $\mathbf{C'_B}$) t-separates **A** and **B** and $\#\mathbf{C'_A} + \#\mathbf{C'_B} \leq r$, then for any $\mathbf{C_A}$, $\mathbf{C_B}$ there is a fixed parameter structural equation model S with path diagram G that is LA below the sets ($\mathbf{C_A}$; $\mathbf{C_B}$) for **A** and **B** that entails $rank(\text{cov}(\mathbf{A},\mathbf{B})) > r$.

In order to use the Extended Trek Separation theorems, it is necessary to have statistical tests of when rank constraints hold, or equivalently, when the corresponding determinants are zero. Drton & Olkin (2008) describe a statistical test of the rank constraints, that assumes a Normal distribution; however, in practice even when the distributions is non-Normal, the test often performs well. The Wishart test for vanishing tetrad constraints is a special case of this test (and was used in all of the simulations performed.)

There is also a much slower, but asymptotically distribution-free statistical test of rank constraints based on the test developed by Bollen and Ting (Bollen & Ting, 1993).

4 FAITHFULNESS

Let a distribution P be *linearly rank-faithful* to a directed acyclic graph G if every rank-constraint on a sub-covariance matrix that holds in P is entailed by every free-parameter linear structural equation model with path diagram equal to G.

If a distribution is linearly rank-faithful to its causal graph, then it is possible to use the rank-constraints among the observed variables to draw conclusions about the t-separation structure of the causal graph by using the Trek Separation Theorem to identify latent choke sets. For example, given a quartet of variables $\mathbf{V} = \{X_1, X_2, X_3, X_4\}$, if for every partition of **V** into two sets of equal size (e.g. $\mathbf{A} = \{X_1, X_2\}$, $\mathbf{B} = \{X_3, X_4\}$) the rank of cov(**A**,**B**) is 1, this indicates that there are sets $\mathbf{C_A}$ with one member and $\mathbf{C_B} = \varnothing$ such that ($\mathbf{C_A}$;$\mathbf{C_B}$) t-separates(**A**;**B**). By combining this with other rank constraints and partial correlation constraints, it is possible to conclude, e.g. that X_1, X_2, X_3 and X_4 have a single latent common cause (Silva et al. 2006)

In practice, there is no oracle that states whether a given rank constraint holds in a population, so statistical tests of rank constraints are substituted for an oracle. But is the assumption of linear rank-faithfulness reasonable? One justification for the assumption of rank-faithfulness is that the Trek Separation Theorem entails that if there is no pair of sets $\mathbf{C_A}$ and $\mathbf{C_B}$ such that $\#\,\mathbf{C_A} + \#\,\mathbf{C_A} \leq r$, and **A** and **B** are t-separated by ($\mathbf{C_A}$;$\mathbf{C_B}$) then the rank of cov(**A**,**B**) is not linearly entailed to be $\leq r$ for all values of the free parameters of a free parameter structural equation model with path diagram G. Moreover, since cov(**A**,**B**) is a linear function of the covariance matrix among the latents and the covariance matrix of the error terms, and the rank is not linearly entailed to be of rank r or less, it follows that the set of values of free parameters for which rank(cov(**A**,**B**)) $\leq r$ is of Lebesgue measure 0. This fact can be used to demonstrate the pointwise consistency of algorithms that rely on statistical tests of rank-constraints (Silva et al. 2006) under the assumption of linear rank-faithfulness.

This does not settle the practicality of such algorithms on reasonable sample sizes. Since statistical tests of the rank constraints are used to determine whether or not a rank-constraint holds in a population, if the relevant determinants that determine rank are very close to, but not exactly equal to zero, any algorithm relying on statistical tests of rank could be incorrect with high probability unless the sample sizes were unrealistically large. This can occur for example, when some of the correlations between observed indicators are either very close to zero or very close to 1. Nevertheless, simulation tests and real applications are positive evidence that BuildPureClusters works at reasonable sample sizes. For a further discussion of faithfulness assumptions see Spirtes et al. (2001), Robins et al. (2003), Kalisch & Buhlmann (2007), and Uhler et al. (2012).

The concept of linear rank faithfulness can be extended in the following way. If **Φ** is a set of functions that contains the linear functions as a special case, a distribution P is <**Φ**, Θ>-*LA below the sets* $\mathbf{C_A}$, $\mathbf{C_B}$ *for* **A**, **B** *rank faithful* to a directed graph G if every rank constraint that holds in P is entailed to hold by every free parameter SEM <**Φ**, Θ> with path diagram G that is LA below the sets $\mathbf{C_A}$, $\mathbf{C_B}$ *for* **A**, **B**.

Suppose in what follows that a given free parameter structural equation model S = <**Φ**, **Θ**> is LA below the sets ($\mathbf{C_A}$; $\mathbf{C_B}$) for **A**, **B**, and that for each equation $X = f(\mathbf{Y})$ in **Φ** not required to be linear by definition, a linear equation with any value of the coefficients

$$X = \sum_{Y \in \mathbf{Y}} a_{X,Y} Y$$

is the result of a substitution of some value for the free parameters in S. For example, if $X = a_1Y + a_2Y^2$, then for any value of a_1, $X = a_1Y$ is the result of setting the free parameter a_2 to zero. In contrast, if Φ contained only $X = a_2Y^2$, the correlations between X and Y would be forced to be zero for all a_2, which in general could lead to rank constraints holding for all values of the free parameters even without the corresponding t-separation relations holding in G.

If all of the variables in **Φ** are analytic functions, whenever the set of solutions to an analytic function is not the entire space of values, the set of solutions has Lebesgue measure 0 (Kilmer et al. 1996). So the same kind of argument for faithfulness in the LA-below-the-choke-set case can be made as in the linear case, as long as **Φ** contains all LA functions among the part of the graph that is not below the choke sets as a special case.

This still leaves the question of whether there are common "almost" violations of rank faithfulness that could only be discovered with enormous sample sizes (i.e. the relevant determinants are very close to zero).

In order to illustrate one use of the extension of the Trek Separation Theorem and to do a preliminary test of the extent to which the introduction of non-linearity makes the problem of almost violations of the assumption of rank-faithfulness more common, I performed a simulation study of the Silva et al. BuildPureClusters Algorithm, using both linear models, and LA-below the choke set models.

The BuildPureClusters Algorithm (Silva et al. 2006) takes as input sample data and attempts to find a subset **S** of the measured indicators such no two members of **S** have a directed edge between them, no member of **S** has more than one latent parent, and the measured variables in **S** are partitioned into clusters, where each member of a cluster is the child of the same latent parent. (This is useful for determining which measured variables are measuring which latent variables, and is input to the MIMBuild algorithm that searches for the causal structure among the latent variables.) BuildPureClusters uses tests of vanishing tetrad differences to select and cluster the variables (which are equivalent to tests of whether various 2×2 submatrices of the covariance matrix have rank 1.) Not all of the rank tests that BuildPureClusters uses in general are also entailed for the case where the relationships between the latents are non-linear (which is *not* the same as LA below the choke sets), but all of the ones that it uses for this particular study are entailed in the non-linear case.

Figure 2 illustrates a model that contains an impure measurement model because of the $X_1 \rightarrow X_6$ edge and because X_{10} has two latent parents (indicated by the red arrows) while Figure 3 illustrates that if X_6 and X_{10} are removed, the resulting model has a pure measurement model. Thus correct output for Figure 2 would either be $\{X_1, X_2, X_3, X_4, X_5\}$ and $\{X_7, X_8, X_9\}$ or $\{X_2, X_3, X_4, X_5\}$ and $\{X_6, X_7, X_8, X_9\}$.

The model in the simulation contained 5 latent variables (L_1 through L_5), each with 5 measured children (X_1 through X_{25}), with L_2 through L_5 children of L_1. It also contained edges $X_1 \rightarrow X_6$, $X_{15} \rightarrow X_{19}$, $L_3 \rightarrow X_{10}$, and $L_4 \rightarrow X_{21}$, which introduced impurities. The input to the algorithm in each case was raw data at one of 3 sample sizes, 100, 500, and 1000. Each variable is a linear function of its parents plus a unique error term, where the linear coefficients were chosen uniformly from the range 0.5 to 2.0, and the error terms were independent standard Gaussian. Each latent variable L_i (i = 2…5) was equal to $aL_1 + bcL_1^3 + \varepsilon_i$, where a was chosen uniformly from 0.25 to 1.0, c was chosen uniformly from 0.5 to 2.0, and the degree of non-linearity was varied by setting b to each of the values 0.0, 0.01, 0.02, 0.03, and 0.05 in turn. The degree of non-linearity of the relationship between the measured variables was measured by the median p-value of the White test of non-linearity between each pair of measured variables (which is 0.5 for linear relationships).

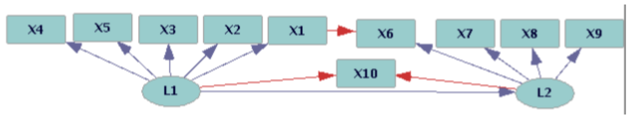


Figure 2: Impure Measurement Model

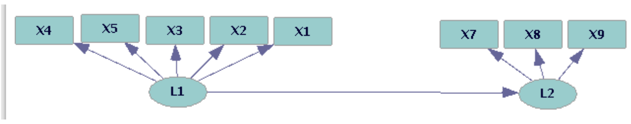


Figure 3: Pure Measurement Submodel

In order to avoid detectible cases of almost unfaithful rank constraints, if the correlation matrix of the observed indicators contained correlations close to zero (less than 0.09) or close to 1 (greater than .9) the data was rejected. (In actual practice, instead of rejecting the data a user could simply search for a subset of variables that did not contain extreme correlations.) The simulation set the p-value used in the algorithm to 0.01 for every case, and the TETRAD IV implementation was used (http://www.phil.cmu.edu/projects/tetrad/current.html).

The correctness of the output of BuildPureClusters was measured in three ways:

1. How many clusters the algorithm found (which was a maximum of 5 in each case).
2. How far a given output cluster was from being a pure cluster. I set Purity for a given output cluster to Purity = (Size of largest pure subcluster contained in output cluster)/(size of output cluster). For example, if the output cluster for data generated by the a model had 7 variables $\{X_1, X_2, X_3, X_4, X_5, X_6, X_9\}$, and $X_1 - X_6$ were all children of latent variable L_1, and X_9 was a child of latent variable L_2, X_9 would have to be removed in order to make the output cluster pure (leaving 6 variables), so Purity for the output cluster would be equal to 6/7.
3. The percentage of the largest pure actual subclusters included in the output. I set Fraction size = (Size of the output cluster)/(size of the largest actual pure subcluster containing it). For example, if a model has an actual pure subcluster of size 8 (e.g. $X_1 - X_8$) and if the output contained a corresponding subcluster of size 6 (e.g. $X_1 - X_6$) then Fraction size for the output cluster is 6/8. (If the output contained only four subclusters instead of the potential five subclusters, I calculated this only for the subclusters that were actually output.

100 data sets were generated at each sample size, for both the linear and non-linear case. Then the BuildPureClusters algorithm was applied to each data set, using the Wishart test of vanishing tetrad differences. The Wishart test assumes the variables have joint Gaussian distributions, which is true in the linear Gaussian case but not the non-linear case.

Size	Cubic	Cluster Number	Average Purity	Average Fraction	Median White
100	0.00	3.89	.909	.782	.500
100	0.01	4.26	.930	.792	.414
100	0.02	4.32	.931	.806	.291
100	0.03	4.26	.935	.809	.285
100	0.05	4.29	.937	.809	.241
500	0.00	4.34	.957	.820	.508
500	0.01	4.41	.953	.813	.349
500	0.02	4.34	.950	.813	.119
500	0.03	4.29	.954	.813	.088
500	0.05	4.48	.957	.829	.0001
1000	0.00	4.78	.930	.900	.510
1000	0.01	4.91	.953	.926	.288
1000	0.02	4.55	.924	.909	.030
1000	0.03	4.31	.912	.899	.017
1000	0.05	4.52	.956	.831	$4.43e^{-10}$

Table 1: Output of First Simulation Study

The results of the simulation test are summarized in Table 1, where each row gives values for 100 runs of a given kind. The columns in order give the sample size, the value of the b coefficient, the average number of clusters, average Purity, average Fraction size, and median p-value of a White test of non-linearity applied to each pair of measured variables. The maximum correct number of clusters is 5, the maximum average Purity is 1, and the maximum average Fraction is 1.

The results of the simulation study (Table 1) indicate that at least with respect to vanishing tetrad differences, the BuildPureClusters algorithm performs about as well in the nonlinear and the linear case using the Wishart test. There is no systematic advantage of linear over non-linear or vice-versa, and the results are generally close in both cases. Hence, in this limited test, the non-linearity that was introduced did not make the problem of almost unfaithful rank constraints much worse in terms of the output.

A simulation study of the extent to which violation of the assumption that the observed variables are linearly related to their latent parents affects the performance of the BuildPureClusters algorithm was also performed. The input to the algorithm in each case was raw data at one of 2 sample sizes, 100 and 1000. The latent variables were simulated in the same was as in the previously described simulation. Each measured variable was set equal to $(1 - d)eL_i + dfL_i^3 + \varepsilon_i$ (where L_i is the parent of the measured variable in the graph), e and f were independently selected from a uniform distribution between 0.5 and 2.0, and the degree of non-linearity of the relationship between the measured and the latents was varied by setting *d* to either 0.01 or 0.05 in turn. The error terms were independent standard Gaussians. The results are shown in Table 2, where the second column reports the values both of *b* (the first number, from the equation for the relationships between the latents) and *d* (the second number, from the equations relating the measured variables to their latent parent.)

As with the previously described simulation, the result of making the relationships between the latents non-linear does not have any systematic effect on the performance of the BuildPureClusters Algorithm However, as the nonlinearity of the relationship between the measured variables and their latent parents increases, the output becomes much less informative (as evidenced by the large decreases in the Number of Clusters, and the Average Fraction), but is generally not incorrect (as evidenced by the small decreases in the Average Purity). When the assumption of linear relationships between the measured and latent variables is violated, the algorithm actually performs better at smaller sample sizes, presumably because at larger sample sizes even small deviations from the assumption lead to rejection of the rank constraints.

Sample Size	Cubic	Number of Clusters	Average Purity	Average Fraction	Median White
100	0:.01	3.42	.909	.755	.302
100	0:.05	2.65	.874	.668	.205
100	.05:.01	2.65	.903	.754	.346
100	.05:.05	3.23	.902	.679	.212
1000	0:.01	2.21	.942	.713	.019
1000	0:.05	0.72	.868	.344	6.1e^{-4}
1000	.05:.01	3.22	.942	.749	.106
1000	.05:.05	1.20	.895	.305	6.9e^{-4}

Table 2: Output of Second Simulation Study

5 OPEN QUESTIONS

In this paper I have proved that the necessary and sufficient conditions for a class of rank constraints on submatrices of a covariance matrix to be implied by a linear model can be extended to models that contains some non-linear and/or cyclic relationships. This shows that existing algorithms that use these rank constraints to search for causal models can be reliably applied to a much wider class of models, as long a rank-faithfulness condition holds. I also argued that the same kind of considerations that argue for rank-faithfulness in linear models can be extended to some kinds of non-linear structure equation models.

In order to make full use of this theorem, it would be very helpful to have a computationally feasible test of when two models are equivalent with respect to rank constraints of a given kind assuming they are both LA below their choke sets. Nor is it known how to graphically represent the features that each member of such an equivalence class has in common. In addition, the question of the extent to which almost violations of faithfulness are made worse by different classes of non-linear functional relationships among variables also needs to be more fully investigated.

6 APPENDIX

The proof of Theorem 2 requires the following two lemmas.

Lemma 1: Suppose that $\mathbf{C_A} \neq \emptyset$, $\mathbf{A} = \mathbf{\Lambda_A C_A} + \mathbf{f}(\mathbf{E_A})$, and $\text{cov}(\mathbf{f}(\mathbf{E_A}),\mathbf{B}) = 0$, where $\mathbf{\Lambda_A}$ is a #**A** by #$\mathbf{C_A}$ matrix of real numbers. Then $rank(\text{cov}(\mathbf{A},\mathbf{B})) \leq \#\mathbf{C_A}$.

Proof.

$\text{cov}(\mathbf{A},\mathbf{B}) = \text{cov}(\mathbf{\Lambda_A C_A} + \text{f}(\mathbf{E_A}),\mathbf{B}) = \text{cov}(\mathbf{\Lambda_A C_A},\mathbf{B}) + \text{cov}(\text{f}(\mathbf{E_A}),\mathbf{B}) = \mathbf{\Lambda_A}\text{cov}(\mathbf{C_A},\mathbf{B})$. Hence $rank(\text{cov}(\mathbf{A},\mathbf{B}) = rank(\mathbf{\Lambda_A}\text{cov}(\mathbf{C_A},\mathbf{B}))$. It follows that

$$
\begin{aligned}
&rank(\text{cov}(\mathbf{C_A},\mathbf{B})) \leq \min(\#\mathbf{C_A},\#\mathbf{B}) \leq \#\mathbf{C_A} \\
&rank(\mathbf{\Lambda_A}) \leq \min(\#\mathbf{C_A},\#\mathbf{A}) \leq \#\mathbf{C_A} \\
&rank(\mathbf{\Lambda_A}\,\text{cov}(\mathbf{C_A},\mathbf{B})) \leq \\
&\min(rank(\mathbf{\Lambda_A}),rank(\text{cov}(\mathbf{C_A},\mathbf{B}))) \leq \\
&\min(\#\mathbf{C_A},\#\mathbf{C_A}) \leq \#\mathbf{C_A}
\end{aligned}
$$

Q.E.D.

Next consider the case where **A** is a linear function of $\mathbf{C_A}$ plus a function of a set of variables $\mathbf{E_A}$, **B** is a linear function of $\mathbf{C_B}$ plus a function of a set of variables $\mathbf{E_B}$, and all of the variables in $\mathbf{E_A}$ are uncorrelated with all of the variables in $\mathbf{E_B}$.

Lemma 2: Suppose that $\mathbf{C_A} \neq \emptyset$, $\mathbf{C_B} \neq \emptyset$, $\#\mathbf{C_A} = p$, $\#\mathbf{C_B} = q$, $\#\mathbf{A} = r$, $\#\mathbf{B} = s$, $\mathbf{A} = \mathbf{\Lambda_A C_A} + \mathbf{f}(\mathbf{E_A})$, $\text{cov}(\mathbf{f}(\mathbf{E_A}), \mathbf{g}(\mathbf{E_B}) = 0$, $\mathbf{B} = \mathbf{\Lambda_B C_B} + \mathbf{f}(\mathbf{E_B})$. Then $rank(\text{cov}(\mathbf{A},\mathbf{B})) \leq \#\mathbf{C_A} + \#\mathbf{C_B}$.

Proof.

$$
\begin{aligned}
&\text{cov}(\mathbf{A},\mathbf{B}) = \text{cov}(\mathbf{\Lambda_A C_A} + \mathbf{f}(\mathbf{E_A}),\mathbf{\Lambda_B C_B} + \mathbf{g}(\mathbf{E_B})) = \\
&\text{cov}(\mathbf{\Lambda_A C_A},\mathbf{\Lambda_B C_B} + \mathbf{g}(\mathbf{E_B})) + \text{cov}(\mathbf{f}(\mathbf{E_A}),\mathbf{\Lambda}_B \mathbf{C_B}) + \\
&\text{cov}(\mathbf{f}(\mathbf{E_A}),\mathbf{g}(\mathbf{E_B})) = \\
&\mathbf{\Lambda_A}\,\text{cov}(\mathbf{C_A},\mathbf{\Lambda_B C_B} + \mathbf{g}(\mathbf{E_B})) + \text{cov}(\mathbf{f}(\mathbf{E_A}),\mathbf{C_B})\mathbf{\Lambda_B}^T
\end{aligned}
$$

$$
\begin{aligned}
&rank(\text{cov}(\mathbf{C_A},\mathbf{\Lambda_B C_B} + \mathbf{g}(\mathbf{E_B}))) \leq \min(p,s) \leq p \\
&rank(\mathbf{\Lambda_A}) \leq \min(r,p) \leq p
\end{aligned}
$$

$$
\begin{aligned}
&rank(\mathbf{\Lambda_A}\,\text{cov}(\mathbf{C_A},\mathbf{\Lambda_B C_B} + \mathbf{g}(\mathbf{E_B}))) \leq \\
&\min(rank(\mathbf{\Lambda_A}),rank(\text{cov}(\mathbf{C_A},\mathbf{\Lambda_B C_B} + \mathbf{g}(\mathbf{E_B}))) \leq \\
&\min(p,p) \leq p \\
&rank(\text{cov}(\mathbf{f}(\mathbf{E_A}),\mathbf{C_B})) \leq \min(r,q) \leq q \\
&rank(\mathbf{\Lambda_B}) \leq \min(q,s) \leq q \\
&rank(\text{cov}(f(\mathbf{E_A}),\mathbf{C_B}))\mathbf{\Lambda_B}^T) \leq \min(q,q) \leq q
\end{aligned}
$$

It follows that the sum of two matrices of the same number of rows and columns is at most $\#\mathbf{C_A}+\#\mathbf{C_B}$. Q.E.D.

Theorem 2: Suppose G is a directed graph containing $\mathbf{C_A}$, $\mathbf{A}$, $\mathbf{C_B}$, and $\mathbf{B}$, and ($\mathbf{C_A}$; $\mathbf{C_B}$) t-separates $\mathbf{A}$ and $\mathbf{B}$ in G. Then for all covariance matrices entailed by a fixed parameter structural equation model S with path diagram G that is LA below the sets $\mathbf{C_A}$ and $\mathbf{C_B}$ for $\mathbf{A}$ and $\mathbf{B}$, $rank(\text{cov}(\mathbf{A},\mathbf{B})) \leq \#\mathbf{C_A} + \#\mathbf{C_B}$.

Proof. In the proof, the graphical relations all refer to G_{ext}, so the graphical arguments will be dropped when referring to parents and directed paths. I will prove the theorem by showing that t-separation of $\mathbf{A}$ and $\mathbf{B}$ by ($\mathbf{C_A}$; $\mathbf{C_B}$) entails that $\mathbf{A}$ can be written as a linear function of $\mathbf{C_A}$ plus a function of a set of variables $\mathbf{E_A}$, that $\mathbf{B}$ can be written as a linear function of $\mathbf{C_B}$ plus a function of a set of variables $\mathbf{E_B}$, and that all of the variables in $\mathbf{E_A}$ are uncorrelated with all of the variables in $\mathbf{E_B}$. Then applying Lemmas 1 and/or 2 proves the theorem.

Case 1: If $\mathbf{C_A} = \mathbf{C_B} = \varnothing$, then there are no treks between $\mathbf{A}$ and $\mathbf{B}$. Hence $\mathbf{A}$ and $\mathbf{B}$ are jointly independent. It follows that $\text{cov}(\mathbf{A},\mathbf{B}) = 0$, which is of rank $0 = \#\mathbf{C_A}+\#\mathbf{C_B}$

Case 2: $\mathbf{C_A} \neq \varnothing$. I will show that for each $A_i \in \mathbf{A}$,

$$A_i = \sum_{V \in \mathbf{C_A}} a_{i,V} V + f_i(\mathbf{E}_i)$$

where each member of $\mathbf{E_i}$ is not in $D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A}$ and is an ancestor of A_i via some (possibly single-vertex) path that does not intersect $\mathbf{C_A}$.

Case 2a: $A_i \in \mathbf{C_A}$. Set $A_i = 1 \times A_i$, $\mathbf{E_i} = \varnothing$, and $f_i(\mathbf{E_i}) = 0$. Since A_i is in $\mathbf{C_A}$, A_i is a linear function of $\mathbf{C_A}$, and trivially each member of $\mathbf{E_i}$ is not in $D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A}$ and is an ancestor of A_i via some (possibly single-vertex) path that does not intersect $\mathbf{C_A}$.

Case 2b: $A_i \notin \mathbf{C_A}$.

Case 2b(i): $D(\mathbf{C_A}, A_i) = \varnothing$. Set $\mathbf{E_i} = \{A_i\}$, $f_i(\mathbf{E_i}) = A_i$. By assumption, each member of $\mathbf{E_i}$ is not in $D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A}$ and is an ancestor of A_i via some (possibly single-vertex) path that does not intersect $\mathbf{C_A}$.

Case 2bii: $D(\mathbf{C_A}, A_i) \neq \varnothing$. The longest directed path from $\mathbf{C_A}$ to A_i is of finite length. Let $\mathbf{R} = \{V \in \mathbf{Parents}(A_i) \cap (D(\mathbf{C_A}) \cup \mathbf{C_A})\}$. By the assumption of LA below the choke sets $\mathbf{C_A}$, $\mathbf{C_B}$, for $\mathbf{A}$, $\mathbf{B}$,

$$A_i = \sum_{\mathbf{R}} a_{i,V} V + f_i(\mathbf{Parents}(A_i) \setminus (D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A}))$$

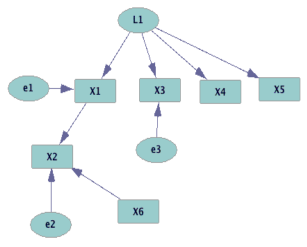
The algorithms in this section of the proof are illustrated in Figure 4 and Figure 5 (where only the relevant error variables are shown in the graph). For each vertex in $\mathbf{R}$, substitute the r.h.s of equation 1 in for V. Continue substitutions until no more substitutions based on equation 1 can be made. The proof is by induction on the number of substitutions.

Let $\mathbf{V_i} = \mathbf{Parents}(A_i) \cap D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A})$, $f_i^1 = f_i$ and $\mathbf{E}_i^1 = \mathbf{Parents}(A_i) \setminus (D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A})$ at stage 1 of equation 1. Every member of $\mathbf{E}_i^1$ is not in $D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A}$ by definition. An edge from any member of $\mathbf{E}_i^1$ to A_i constitutes a path to A_i that does not intersect $\mathbf{C_A}$.

Suppose for an induction hypothesis that after n substitutions,

$$A_i = \sum_{V \in \mathbf{V}_n} a^n_{i,V} V + f_i^n(\mathbf{E}_i^n) + a^n_{i,X} X$$

where $\mathbf{V}_n \subseteq D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A}$, $X \in D(\mathbf{C_A}, \mathbf{A}) \cap \mathbf{V}_n$, there is no member of $\mathbf{V}_n$ whose longest path to A_i is shorter than the longest path from X to A_i, and each member of $\mathbf{E}_i^n$ is not a member of $D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A}$ but is an ancestor of A_i via a directed path that does not intersect $\mathbf{C_A}$. The superscripts represent which substitution the superscripted term first appeared in. If no such X exists (because A_i is expressed as a function of members of $\mathbf{C_A}$ and variables that are not on paths from $\mathbf{C_A}$ to $\mathbf{A}$), the algorithm is done.



$X_2 = 3\,X_1 + f_2(\varepsilon_2, X_6)$ $X_4 = 0.6\,L_1 + f_4(\varepsilon_4)$
$X_1 = 2\,L_1 + f_1(\varepsilon_1)$ $X_5 = 0.9\,L_1 + f_5(\varepsilon_5)$
$X_3 = 0.8\,L_1 + f_3(\varepsilon_3)$
$\mathbf{A} = \{X_2, X_3\}$ $\mathbf{B} = \{X_4, X_5\}$ $\mathbf{C_A} = \{L_1\}$ $\mathbf{C_B} = \varnothing$
$D(\mathbf{C_A}, \mathbf{A}) = \{X_1, X_2, X_3\}$ $D(\mathbf{C_B}, \mathbf{B}) = \varnothing$

Figure 4: Illustration of base stage of substitutions

Otherwise, Let $\mathbf{R} = \mathbf{Parents}(X) \cap D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A}$. Substitute the r.h.s. of

$$X = \sum_{V \in \mathbf{R}} a_{X,V} V + f_X(\mathbf{Parents}(X) \setminus (D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A})$$

in for X in the equation. After the substitution,

$$A_i = \sum_{V \in \mathbf{V}_n} a^n_{i,V} V + f_i^n(\mathbf{E}_i^n) +$$

$$a^n_{i,X} \left(\sum_{V \in \mathbf{V}_n} a_{X,V} V + f_x(\mathbf{Parents}(X) \setminus (D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A})) \right)$$

$$= \sum_{V \in \mathbf{V}_n} a^n_{i,V} V + a^n_{i,X} \left(\sum_{V \in \mathbf{R}} a_{X,V} V \right) +$$

$$f_i^n(\mathbf{E}_i^n) + a^n_{i,X} f(\mathbf{Parents}(X) \setminus (D(\mathbf{C_A}, \mathbf{A}) \cup \mathbf{C_A})) =$$

$$\sum_{V\in\mathbf{V}_{n+1}} a^{n+1}_{i,V}V + f^{n+1}_i(\mathbf{E}^{n+1}_i),$$
$$\mathbf{V}_{n+1} = (\mathbf{V}_n \cup (\mathbf{Parents}(X) \cap (D(\mathbf{C_A},\mathbf{A}) \cup \mathbf{C_A}))) \setminus X$$
$$\mathbf{E}^{n+1}_i = \mathbf{E}^n_i \cup (\mathbf{Parents}(X) \setminus (D(\mathbf{C_A},\mathbf{A}) \cup \mathbf{C_A})),$$
$$a^{n+1}_{i,V} = a^n_{i,X} a_{X,V} \text{ for parents of } X$$
$$f^{n+1}_i(\mathrm{E}^{n+1}_i) = f^n_i(\mathrm{E}^n_i) +$$
$$a_{i,X} f^n_x(\mathbf{Parents}(X) \setminus (D(\mathbf{C_A},\mathbf{A}) \cup \mathbf{C_A}))$$

Figure 5 contains an illustration of the substitutions for the example shown in Figure 4.

Each member of $\mathbf{E}^{n+1}_i \cap \mathbf{E}^n_i$ is not on a directed path from $\mathbf{C_A}$ to $\mathbf{A}$ but is an ancestor of A_i via a path that does not intersect $\mathbf{C_A}$ by the induction assumption. $\mathbf{E}^{n+1}_{A'} \setminus \mathbf{E}^n_{A'} \subseteq \mathbf{Parents}(X) \setminus (D(\mathbf{C_A},\mathbf{A}) \cup \mathbf{C_A})$ and hence not a member of $D(\mathbf{C_A},\mathbf{A}) \cup \mathbf{C_A}$. Because X is expanded by substitution only if it is not in $\mathbf{C_A}$, and occurs on the r.h.s only by substituting in for variables not in $\mathbf{C_A}$, X is an ancestor of A_i via a directed path that does not intersect $\mathbf{C_A}$; hence each member of $\mathbf{Parents}(X) \setminus (D(\mathbf{C_A},\mathbf{A}) \cup \mathbf{C_A})$ is an ancestor of A_i via a directed path that doesn't intersect $\mathbf{C_A}$.

$X_2 = 3\,X_1 + f_2(\varepsilon_2, X_6)$ $\quad \mathbf{V_1} = \{X_1\}$

$\mathbf{E}^1_2 = \{X_6, \varepsilon_2\}$ $\quad f^1_2(\varepsilon_2, X_6) = f_2(\varepsilon_2, X_6)$ $\quad a^1_{X_2,X_1} = 3$

Substitute r.h.s of equation for X_1 in for X_1, in equation for X_2

$X_2 = 3\,X_1 + f_2(\varepsilon_2, X_6) = 3\,(2\,L_1 + f_1(\varepsilon_1)) + f_2(\varepsilon_2, X_6) = 6\,L_1 + 3\,f_1(\varepsilon_1)) + f_2(\varepsilon_2, X_6)$

$\mathbf{V_2} = \{L_1\}$

$\mathbf{E}^2_2 = \{X_6, \varepsilon_2, \varepsilon_1\}$ $\quad f^2_2(X_6, \varepsilon_2, \varepsilon_1) = f^1_2(X_6, \varepsilon_2) + 3f^1_2(\varepsilon_1)$

$a^2_{X_2,L_1} = 3 \times 2$ $\quad \mathbf{E_A} = \{X_6, \varepsilon_1, \varepsilon_3\}$

Figure 5: Illustration of substitutions

After a finite number of substitutions, all of the members of $\mathbf{V_n}$ are members of $\mathbf{C_A}$, and no more substitutions are done. At that stage, by induction,

$$A_i = \sum_{V\in\mathbf{C_A}} a_{i,V}V + f(\mathbf{E}_i)$$

where $\mathbf{E}_i \cap (D(\mathbf{C_A},\mathbf{A}) \cup \mathbf{C_A}) = \varnothing$, but each member of E_i is an ancestor of A_i via some path that does not intersect $\mathbf{C_A}$.

Case 2b(ii): $D(\mathbf{C_A}, A_i) \neq \varnothing$. This case now divides into two subcases, $\mathbf{C_B} = \varnothing$ or $\mathbf{C_B} \neq \varnothing$. First consider the case where $\mathbf{C_B} = \varnothing$. Let $\mathrm{E}_\mathbf{A}$ be the union of all of the E_i. For each $X \in \mathbf{E}_i$, if there is a trek T between X and $\mathbf{B}$, then it intersects $\mathbf{C_A}$ on the X side, since otherwise ($\mathbf{C_A}$; $\varnothing$) does not t-separate $\mathbf{A}$ and $\mathbf{B}$. It follows then that there is a directed path from $\mathbf{C_A}$ to X, and X is on a directed path from $\mathbf{C_A}$ to $\mathbf{A}$, contrary to what was proved about each member of $\mathbf{E_A}$. Hence there is no trek between X and $\mathbf{B}$. It follows that $\mathbf{E_A}$ is independent of $\mathbf{B}$, and hence $\mathbf{f}(\mathbf{E_A})$ is independent of $\mathbf{B}$, and cov($\mathbf{f}(\mathbf{E_A})$,$\mathbf{B}$) = 0. Then by Lemma 1, *rank*(cov($\mathbf{A}$,$\mathbf{B}$)) $\leq$ #$\mathbf{C_A}$.

Now suppose $\mathbf{C_B} \neq \varnothing$. Similarly to the case for $\mathbf{A}$, for each B_i in $\mathbf{B}$,

$$B_i = \sum_{V\in\mathbf{C_B}} b_{i,V}V + g_i(\mathbf{E}_i)$$

where each member of $\mathbf{E}_i$ is not in $D(\mathbf{C_B},\mathbf{B}) \cup \mathbf{C_B}$, but is an ancestor of B_i via some path that does not intersect $\mathbf{C_B}$.

I will now show that for any two members X and Y of $\mathbf{E_A}$ and $\mathbf{E_B}$ respectively, cov(X,Y) = 0. By the construction of $\mathbf{E_A}$ and $\mathbf{E_B}$, there is a directed paths P_1 from X to some A_i that does not intersect $\mathbf{C_A}$, and a directed path P_2 from Y to some B_j that does not intersect $\mathbf{C_B}$. If $X = Y$, then there is a trek between $\mathbf{A}$ and $\mathbf{B}$ that does not intersect $\mathbf{C_A}$ on the $\mathbf{A}$ side or $\mathbf{C_B}$ on the $\mathbf{B}$ side, contrary to the assumption that ($\mathbf{C_A}$;$\mathbf{C_B}$) t-separates $\mathbf{A}$ and $\mathbf{B}$. Similarly, if $X \neq Y$, but there is a trek T between X and Y, it either intersects $\mathbf{C_A}$ on the X side or $\mathbf{C_B}$ on the Y side, since otherwise ($\mathbf{C_A}$;$\mathbf{C_B}$) does not t-separate $\mathbf{A}$ and $\mathbf{B}$. But if T intersects $\mathbf{C_A}$ on the X side or $\mathbf{C_B}$ on the Y side, then there is a directed path from $\mathbf{C_A}$ to X or $\mathbf{C_B}$ to Y, in which case X is on a directed path from $\mathbf{C_A}$ to $\mathbf{A}$, or Y is on a directed path from $\mathbf{C_B}$ to $\mathbf{B}$, contrary to what was shown about $\mathbf{E_A}$ and $\mathbf{E_B}$. Hence there is no trek between X and Y and $X \neq Y$. It follows that $\mathbf{E_A}$ is independent of $\mathbf{E_B}$, and for any functions $\mathbf{f}$ and $\mathbf{g}$, $\mathbf{f}(\mathbf{E_A})$ is independent of $\mathbf{g}(\mathbf{E_B})$. Hence cov($\mathbf{f}(\mathbf{E_A})$, $\mathbf{g}(\mathbf{E_B})$) = 0. By Lemma 2, *rank*(cov($\mathbf{A}$,$\mathbf{B}$)) $\leq$ #$\mathbf{C_A}$ + #$\mathbf{C_B}$. Q.E.D.

Theorem 3: For all directed graphs G, if there does not exist a pair of sets $\mathbf{C'_A}$, $\mathbf{C'_B}$, such that ($\mathbf{C'_A}$; $\mathbf{C'_B}$) t-separates $\mathbf{A}$ and $\mathbf{B}$ and #$\mathbf{C'_A}$ + #$\mathbf{C'_B}$ $\leq r$, then for any $\mathbf{C_A}$, $\mathbf{C_B}$ there is a fixed parameter structural equation model S with path diagram G that is LA below the sets ($\mathbf{C_A}$; $\mathbf{C_B}$) for $\mathbf{A}$ and $\mathbf{B}$ that entails *rank*(cov($\mathbf{A}$,$\mathbf{B}$)) > r.

Proof. G can always be made acyclic by setting the coefficients of edges occurring in cycles to zero. By the Trek Separation Theorem, there is a fixed parameter linear structural equation model S' with path diagram G in which *rank*(cov($\mathbf{A}$,$\mathbf{B}$)) > r. By definition, S' is LA below the sets $\mathbf{C_A}$, $\mathbf{C_B}$ for any $\mathbf{C_A}$, $\mathbf{C_B}$. Q.E.D.

Acknowledgements: I wish to thank Rina Foygel, Kelli Talaska, Jan Draisma, Seth Sullivant, and Mathias Drton for substantial help with the proofs at a 2010 AIM workshop on Parameter Identification in Graphical Models.

REFERENCES

Bartholomew, D. J., Steele, F., Moustaki, I., & Galbraith, J. I. (2002). *The Analysis and Interpretation of Multivariate Data for Social Scientists* (Texts in Statistical Science Series). Chapman & Hall/CRC.

Bollen, K. A. (1989). *Structural Equations with Latent Variables.* Wiley-Interscience.

Drton, M., Massam, H., & Olkin, I. (2008). Moments of minors of Wishart matrices. *Annals of Statistics* **36**(5): 2261-2283

Elidan, G., Lotner, N., Friedman, N., & Koller, D. (2001). Discovering hidden variables: A structure-based approach. *Proceedings from Advanced in Neural Information Processing Systems.*

Harman, H. H. (1976). *Modern Factor Analysis*. University Of Chicago Press.

Kalisch, M., and P. Bühlmann (2007). Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, **8**, 613–636.

Kilmer, S., Light, W., Sun, X. & Yu, X. (1996) Approximation by Translates of a Positive Definite Function, *J. Mathematical Analysis and Applications,* **201**, 631-641.

Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press.

Robins, J., Scheines, R., Spirtes, P. & Wasserman, L. (2003) Uniform Consistency In Causal Inference, *Biometrika*, September, 2003, **90**, 491-515.

Jackson, A., and Scheines, R. (2005) Single Mothers' Self-Efficacy, Parenting in the Home Environment, and Children's Development in a Two-Wave Study, in *Social Work Research*, **29**, 1, pp. 7-20.

Silva, R., Scheines, R., Glymour, C., & Spirtes, P. (2006). Learning the structure of linear latent variable models. *J Mach Learn Res*, **7**, 191-246.

Spirtes, P. (1995) Directed Cyclic Graphical Representation of Feedback Models, in *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, ed. by Philippe Besnard and Steve Hanks, Morgan Kaufmann Publishers, Inc., San Mateo.

Spirtes, P., Meek, C., & Richardson, T. S. (1995). Causal inference in the presence of latent variables and selection bias. *Proceedings from Eleventh Conference on Uncertainty in Artificial Intelligence*, San Francisco, CA.

Spirtes, P., Glymour, C., & Scheines, R. (2001). *Causation, Prediction, and Search*, Second Edition (Adaptive Computation and Machine Learning). The MIT Press.

Sullivant, S., Talaska, K., & Draisma, J. (2010). Trek Separation for Gaussian Graphical Models. **Ann Stat**, *38*(3), 1665-1685.

Thurstone, L. (1936). *The Vectors Of Mind: Multiple Factor Analysis For The Isolation Of Primary Traits*. Nabu Press.

Uhler, C. Raskutti, G., Buhlmann, P, and Yu, B. (2012). Geometry of faithfulness assumption in causal inference, Arxiv.Org Math 1207.0547, to appear in *Annals Of Statistics*.

Modeling Documents with a Deep Boltzmann Machine

Nitish Srivastava **Ruslan Salakhutdinov** **Geoffrey Hinton**
{nitish, rsalakhu, hinton}@cs.toronto.edu
Department of Computer Science, University of Toronto
Toronto, Ontario, M5S 3G4 Canada.

Abstract

We introduce a type of Deep Boltzmann Machine (DBM) that is suitable for extracting distributed semantic representations from a large unstructured collection of documents. We overcome the apparent difficulty of training a DBM with judicious parameter tying. This enables an efficient pretraining algorithm and a state initialization scheme for fast inference. The model can be trained just as efficiently as a standard Restricted Boltzmann Machine. Our experiments show that the model assigns better log probability to unseen data than the Replicated Softmax model. Features extracted from our model outperform LDA, Replicated Softmax, and DocNADE models on document retrieval and document classification tasks.

1 Introduction

Text documents are a ubiquitous source of information. Representing the information content of a document in a form that is suitable for solving real-world problems is an important task. The aim of topic modeling is to create such representations by discovering latent topic structure in collections of documents. These representations are useful for document classification and retrieval tasks, making topic modeling an important machine learning problem.

The most common approach to topic modeling is to build a generative probabilistic model of the bag of words in a document. Directed graphical models, such as Latent Dirichlet Allocation (LDA), CTM, H-LDA, have been extensively used for this [3, 2, 8]. Non-parametric extensions of these models have also been quite successful [13, 1, 5]. Even though exact inference in these models is hard, efficient inference schemes, including stochastic variational inference, online inference, and collapsed Gibbs have been developed that make it feasible to train and use these methods [14, 16, 4]. Another approach is to use undirected graphical models such as the Replicated Softmax model [12]. In this model, inferring latent topic representations is exact and efficient. However, training is still hard and often requires careful hyperparameter selection. These models typically perform better than LDA in terms of both the log probability they assign to unseen data and their document retrieval and document classification accuracy. Recently, neural network based approaches, such as Neural Autoregressive Density Estimators (DocNADE) [7], have been to shown to outperform the Replicated Softmax model.

The Replicated Softmax model is a family of Restricted Boltzmann Machines (RBMs) with shared parameters. An important feature of RBMs is that they solve the "explaining-way" problem of directed graphical models by having a complementary prior over hidden units. However, this implicit prior may not be the best prior to use and having some degree of flexibility in defining the prior may be advantageous. One way of adding this additional degree of flexibility, while still avoiding the explaining-away problem, is to learn a two hidden layer Deep Boltzmann Machine (DBM). This model adds another layer of hidden units on top of the first hidden layer with bi-partite, undirected connections. The new connections come with a new set of weights. However, this additional implicit prior comes at the cost of more expensive training and inference. Therefore, we have the following two extremes: On one hand, RBMs can be efficiently trained (e.g. using Contrastive Divergence), inferring the state of the hidden units is exact, but the model defines a rigid, implicit prior. On the other hand, a two hidden layer DBM defines a more flexible prior over the hidden representations, but training and performing inference in a DBM model is considerably harder.

In this paper, we try to find middle ground between these extremes and build a model that combines the best of both. We introduce a two hidden layer DBM model, which we call the Over-Replicated Softmax model. This model is easy to train, has fast approximate inference and still retains some degree of flexibility towards manipulating the prior. Our experiments show that this flexibility is enough to improve significantly on the performance of the standard Replicated Softmax model, both as generative models and as feature extractors even though the new model only has one more parameter than the RBM model. The model also outperforms LDA and DocNADE in terms of classification and retrieval tasks.

Before we describe our model, we briefly review the Replicated Softmax model [12] which is a stepping stone towards the proposed Over-Replicated Softmax model.

2 Replicated Softmax Model

This model comprises of a family of Restricted Boltzmann Machines. Each RBM has "softmax" visible variables that can have one of a number of different states. Specifically, let K be the dictionary size, N be the number of words appearing in a document, and $\mathbf{h} \in \{0,1\}^F$ be binary stochastic hidden topic features. Let $\mathbf{V}$ be a $N \times K$ observed binary matrix with $v_{ik} = 1$ if visible unit i takes on the k^{th} value. We define the energy of the state $\{\mathbf{V}, \mathbf{h}\}$ as :

$$\begin{aligned} E(\mathbf{V}, \mathbf{h}; \boldsymbol{\theta}) &= -\sum_{i=1}^{N}\sum_{j=1}^{F}\sum_{k=1}^{K} W_{ijk} h_j v_{ik} \qquad (1) \\ &\quad -\sum_{i=1}^{N}\sum_{k=1}^{K} v_{ik} b_{ik} - N\sum_{j=1}^{F} h_j a_j, \end{aligned}$$

where $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{a}, \mathbf{b}\}$ are the model parameters; W_{ijk} is a symmetric interaction term between visible unit i that takes on value k, and hidden feature j, b_{ik} is the bias of unit i that takes on value k, and a_j is the bias of hidden feature j. The probability that the model assigns to a visible binary matrix $\mathbf{V}$ is:

$$\begin{aligned} P(\mathbf{V}; \boldsymbol{\theta}) &= \frac{1}{\mathcal{Z}(\boldsymbol{\theta}, N)} \sum_{\mathbf{h}} \exp\left(-E(\mathbf{V}, \mathbf{h}; \boldsymbol{\theta})\right) \qquad (2) \\ \mathcal{Z}(\boldsymbol{\theta}, N) &= \sum_{\mathbf{V}'}\sum_{\mathbf{h}'} \exp\left(-E(\mathbf{V}', \mathbf{h}'; \boldsymbol{\theta})\right), \end{aligned}$$

where $\mathcal{Z}(\boldsymbol{\theta}, N)$ is known as the partition function, or normalizing constant.

The key assumption of the Replicated Softmax model is that for each document we create a separate RBM with as many softmax units as there are words in the document, as shown in Fig. 1. Assuming that the order of the words can be ignored, all of these softmax units can share the same set of weights, connecting them to binary hidden units. In this case, the energy of the state $\{\mathbf{V}, \mathbf{h}\}$ for a document that contains N words is defined as:

$$E(\mathbf{V}, \mathbf{h}) = -\sum_{j=1}^{F}\sum_{k=1}^{K} W_{jk} h_j \hat{v}_k - \sum_{k=1}^{K} \hat{v}_k b_k - N\sum_{j=1}^{F} h_j a_j,$$

where $\hat{v}_k = \sum_{i=1}^{N} v_i^k$ denotes the count for the k^{th} word. The bias terms of the hidden variables are scaled up by the length of the document. This scaling is important as it allows hidden units to behave sensibly when dealing with documents of different lengths. The conditional distributions are given by softmax and logistic functions:

$$\begin{aligned} P(h_j^{(1)} = 1) &= \sigma\Big(\sum_{k=1}^{K} W_{jk}\hat{v}_k + N a_j\Big), \qquad (3) \\ P(v_{ik} = 1) &= \frac{\exp\left(\sum_{j=1}^{F} W_{jk} h_j^{(1)} + b_k\right)}{\sum_{k'=1}^{K} \exp\left(\sum_{j=1}^{F} W_{jk'} h_j^{(1)} + b_{k'}\right)}. \qquad (4) \end{aligned}$$

The Replicated Softmax model can also be interpreted as an RBM model that uses a single visible multinomial unit with support $\{1, ..., K\}$ which is sampled N times (see Fig. 1, right panel).

For this model, exact maximum likelihood learning is intractable, because computing the derivatives of the partition function, needed for learning, takes time that is exponential in $\min\{D, F\}$, i.e the number of visible or hidden units. In practice, approximate learning is performed using Contrastive Divergence (CD) [6].

3 Over-Replicated Softmax Model

The Over-Replicated Softmax model is a family of two hidden layer Deep Boltzmann Machines (DBM). Let

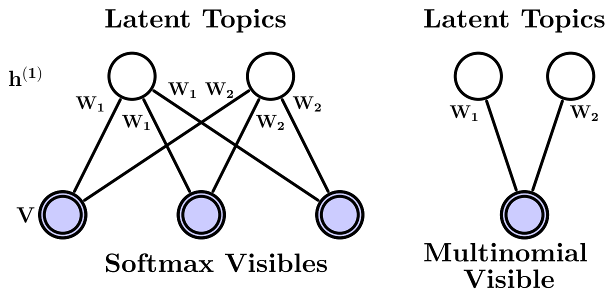


Figure 1: The Replicated Softmax model. The top layer represents a vector $\mathbf{h}$ of stochastic, binary topic features and the bottom layer represents softmax visible units $\mathbf{V}$. All visible units share the same set of weights, connecting them to binary hidden units. **Left:** The model for a document containing three words. **Right:** A different interpretation of the Replicated Softmax model, in which N softmax units with identical weights are replaced by a single multinomial unit which is sampled N times.

us consider constructing a Boltzmann Machine with two hidden layers for a document containing N words, as shown in Fig. 2. The visible layer $\mathbf{V}$ consists of N softmax units. These units are connected to a binary hidden layer $\mathbf{h}^{(1)}$ with shared weights, exactly like in the Replicated Softmax model in Fig. 1. The second hidden layer consists of M softmax units represented by $\mathbf{H}^{(2)}$. Similar to $\mathbf{V}$, $\mathbf{H}^{(2)}$ is an $M \times K$ binary matrix with $h^{(2)}_{mk} = 1$ if the m-th hidden softmax unit takes on the k-th value.

The energy of the joint configuration $\{\mathbf{V}, \mathbf{h}^{(1)}, \mathbf{H}^{(2)}\}$ is defined as:

$$E(\mathbf{V}, \mathbf{h}^{(1)}, \mathbf{H}^{(2)}; \boldsymbol{\theta}) = -\sum_{i=1}^{N}\sum_{j=1}^{F}\sum_{k=1}^{K} W^{(1)}_{ijk} h^{(1)}_j v_{ik} \quad (5)$$
$$-\sum_{i'=1}^{M}\sum_{j=1}^{F}\sum_{k=1}^{K} W^{(2)}_{i'jk} h^{(1)}_j h^{(2)}_{i'k} - \sum_{i=1}^{N}\sum_{k=1}^{K} v_{ik} b^{(1)}_{ik}$$
$$-(M+N)\sum_{j=1}^{F} h^{(1)}_j a_j - \sum_{i=1}^{M}\sum_{k=1}^{K} h^{(2)}_{ik} b^{(2)}_{ik}$$

where $\boldsymbol{\theta} = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{a}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}\}$ are the model parameters.

Similar to the Replicated Softmax model, we create a separate document-specific DBM with as many visible softmax units as there are words in the document. We also fix the number M of the second-layer softmax units across all documents. Ignoring the order of the words, all of the first layer softmax units share the same set of weights. Moreover, the first and second layer weights are tied. Thus we have $W^{(1)}_{ijk} = W^{(2)}_{i'jk} = W_{jk}$ and $b^{(1)}_{ik} = b^{(2)}_{i'k} = b_k$. Compared to the standard Replicated Softmax model, this model has more replicated softmaxes (hence the name "Over-Replicated"). Unlike the visible softmaxes, these additional softmaxes are unobserved and constitute a second hidden layer[1]. The energy can be simplified to:

$$E(\mathbf{V}, \mathbf{h}^{(1)}, \mathbf{H}^{(2)}; \boldsymbol{\theta}) = -\sum_{j=1}^{F}\sum_{k=1}^{K} W_{jk} h^{(1)}_j \left(\hat{v}_k + \hat{h}^{(2)}_k\right) \quad (6)$$
$$-\sum_{k=1}^{K}\left(\hat{v}_k + \hat{h}^{(2)}_k\right) b_k - (M+N)\sum_{j=1}^{F} h^{(1)}_j a_j$$

where $\hat{v}_k = \sum_{i=1}^{N} v_{ik}$ denotes the count for the k^{th} word in the input and $\hat{h}^{(2)}_k = \sum_{i=1}^{M} h^{(2)}_{ik}$ denotes the count for the k^{th} "latent" word in the second hidden layer. The joint probability distribution is defined as:

$$P(\mathbf{V}, \mathbf{h}^{(1)}, \mathbf{H}^{(2)}; \boldsymbol{\theta}) = \frac{\exp\left(-E(\mathbf{V}, \mathbf{h}^{(1)}, \mathbf{H}^{(2)}; \boldsymbol{\theta})\right)}{\mathcal{Z}(\boldsymbol{\theta}, N)},$$

[1]This model can also be seen as a Dual-Wing Harmonium [17] in which one wing is unclamped.

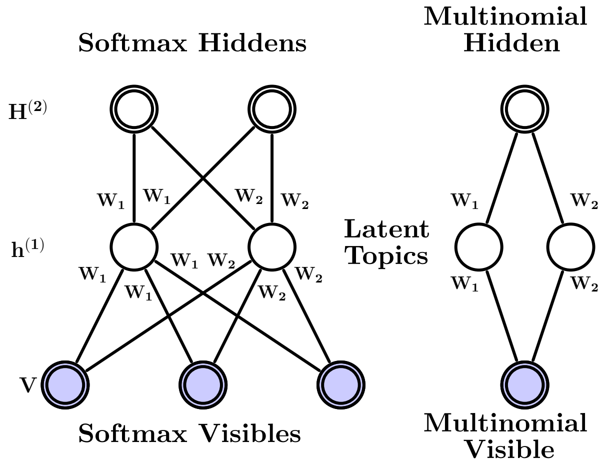


Figure 2: The Over-Replicated Softmax model. The bottom layer represents softmax visible units $\mathbf{V}$. The middle layer represents binary latent topics $\mathbf{h}^{(1)}$. The top layer represents softmax hidden units $\mathbf{H}^{(2)}$. All visible and hidden softmax units share the same set of weights, connecting them to binary hidden units. **Left:** The model for a document containing $N = 3$ words with $M = 2$ softmax hidden units. **Right:** A different interpretation of the model, in which N softmax units with identical weights are replaced by a single multinomial unit which is sampled N times and the M softmax hidden units are replaced by a multinomial unit sampled M times.

Note that the normalizing constant depends on the number of words N in the corresponding document, since the model contains as many visible softmax units as there are words in the document. So the model can be viewed as a family of different-sized DBMs that are created for documents of different lengths, but with a fixed-sized second-layer.

A pleasing property of the Over-Replicated Softmax model is that it has exactly the same number of trainable parameters as the Replicated Softmax model. However, the model's marginal distribution over $\mathbf{V}$ is different, as the second hidden layer provides an additional implicit prior. The model's prior over the latent topics $\mathbf{h}^{(1)}$ can be viewed as the geometric mean of the two probability distributions: one defined by an RBM composed of $\mathbf{V}$ and $\mathbf{h}^{(1)}$, and the other defined by an RBM composed of $\mathbf{h}^{(1)}$ and $\mathbf{H}^{(2)}$:[2]

$$P(\mathbf{h}^{(1)}; \boldsymbol{\theta}) = \frac{1}{\mathcal{Z}(\boldsymbol{\theta}, N)} \underbrace{\left(\sum_{\mathbf{v}} \exp\left(\sum_{j=1}^{F}\sum_{k=1}^{K} W_{jk} \hat{v}_k h^{(1)}_j\right)\right)}_{\text{RBM with } \mathbf{h}^{(1)} \text{ and } \mathbf{v}}$$
$$\underbrace{\left(\sum_{\mathbf{H}^{(2)}} \exp\left(\sum_{j=1}^{F}\sum_{k=1}^{K} W_{jk} \hat{h}^{(2)}_k h^{(1)}_j\right)\right)}_{\text{RBM with } \mathbf{h}^{(1)} \text{ and } \mathbf{H}^{(2)}}. \quad (7)$$

Observe that $\sum_{k=1}^{K} \hat{v}_k = N$ and $\sum_{k=1}^{K} \hat{h}^{(2)}_k = M$, so the strength of this prior can be varied by changing the number M of second-layer softmax units. For example,

[2]We omit the bias terms for clarity of presentation.

if $M = N$, then the model's marginal distribution over $\mathbf{h}^{(1)}$, defined in Eq. 7, is given by the product of two identical distributions. In this DBM, the second-layer performs 1/2 of the modeling work compared to the first layer [11]. Hence, for documents containing few words ($N \ll M$) the prior over hidden topics $\mathbf{h}^{(1)}$ will be dominated by the second-layer, whereas for long documents ($N \gg M$) the effect of having a second-layer will diminish. As we show in our experimental results, having this additional flexibility in terms of defining an implicit prior over $\mathbf{h}^{(1)}$ significantly improves model performance, particularly for small and medium-sized documents.

3.1 Learning

Let $\mathbf{h} = \{\mathbf{h}^{(1)}, \mathbf{H^{(2)}}\}$ be the set of hidden units in the two-layer DBM. Given a collection of L documents $\{\mathbf{V}\}_{l=1}^{L}$, the derivative of the log-likelihood with respect to model parameters W takes the form:

$$\frac{1}{L}\sum_{l=1}^{L}\frac{\partial \log P(\mathbf{V_l};\boldsymbol{\theta})}{\partial W_{jk}} = \mathbb{E}_{P_{\text{data}}}\left[(\hat{v}_k + \hat{h}_k^{(2)})h_j^{(1)}\right] - \mathbb{E}_{P_{\text{Model}}}\left[(\hat{v}_k + \hat{h}_k^{(2)})h_j^{(1)}\right],$$

where $\mathbb{E}_{P_{\text{data}}}[\cdot]$ denotes an expectation with respect to the data distribution $P_{\text{data}}(\mathbf{h},\mathbf{V}) = P(\mathbf{h}|\mathbf{V};\boldsymbol{\theta})P_{\text{data}}(\mathbf{V})$, with $P_{\text{data}}(\mathbf{V}) = \frac{1}{L}\sum_l \delta(\mathbf{V}-\mathbf{V}_l)$ representing the empirical distribution, and $\mathbb{E}_{P_{\text{Model}}}[\cdot]$ is an expectation with respect to the distribution defined by the model. Similar to the Replicated Softmax model, exact maximum likelihood learning is intractable, but approximate learning can be performed using a variational approach [10]. We use mean-field inference to estimate data-dependent expectations and an MCMC based stochastic approximation procedure to approximate the models expected sufficient statistics.

Consider any approximating distribution $Q(\mathbf{h}|\mathbf{V};\boldsymbol{\mu})$, parameterized by a vector of parameters $\boldsymbol{\mu}$, for the posterior $P(\mathbf{h}|\mathbf{V};\boldsymbol{\theta})$. Then the log-likelihood of the DBM model has the following variational lower bound:

$$\log P(\mathbf{V};\boldsymbol{\theta}) \geq \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{V};\boldsymbol{\mu})\log P(\mathbf{V},\mathbf{h};\boldsymbol{\theta}) + \mathcal{H}(Q),$$

where $\mathcal{H}(\cdot)$ is the entropy functional. The bound becomes tight if and only if $Q(\mathbf{h}|\mathbf{V};\boldsymbol{\mu}) = P(\mathbf{h}|\mathbf{V};\boldsymbol{\theta})$.

For simplicity and speed, we approximate the true posterior $P(\mathbf{h}|\mathbf{V};\boldsymbol{\theta})$ with a fully factorized approximating distribution over the two sets of hidden units, which corresponds to the so-called mean-field approximation:

$$Q^{MF}(\mathbf{h}|\mathbf{V};\boldsymbol{\mu}) = \prod_{j=1}^{F} q(h_j^{(1)}|\mathbf{V})\prod_{i=1}^{M} q(h_i^{(2)}|\mathbf{V}), \qquad (8)$$

where $\boldsymbol{\mu} = \{\boldsymbol{\mu}^{(1)}, \boldsymbol{\mu}^{(2)}\}$ are the mean-field parameters with $q(h_j^{(1)} = 1) = \mu_j^{(1)}$ and $q(h_{ik}^{(2)} = 1) = \mu_k^{(2)}$, $\forall i \in \{1,\ldots,M\}$, s.t. $\sum_{k=1}^{K}\mu_k^{(2)} = 1$. Note that due to the shared weights across all of the hidden softmaxes, $q(h_{ik}^{(2)})$ does not dependent on i. In this case, the variational lower bound takes a particularly simple form:

$$\log P(\mathbf{V};\boldsymbol{\theta}) \geq \sum_{\mathbf{h}} Q^{MF}(\mathbf{h}|\mathbf{V};\boldsymbol{\mu})\log P(\mathbf{V},\mathbf{h};\boldsymbol{\theta}) + \mathcal{H}(Q^{MF})$$
$$\geq \left(\hat{\mathbf{v}}^{\top} + M\boldsymbol{\mu}^{(2)\top}\right)\mathbf{W}\boldsymbol{\mu}^{(1)} - \log\mathcal{Z}(\boldsymbol{\theta},N) + \mathcal{H}(Q^{MF}),$$

where $\hat{\mathbf{v}}$ is a $K\times 1$ vector, with its k^{th} element $\hat{v}_k$ containing the count for the k^{th} word. Since $\sum_{k=1}^{K}\hat{v}_k = N$ and $\sum_{k=1}^{K}\mu_k^{(2)} = 1$, the first term in the bound linearly combines the effect of the data (which scales as N) with the prior (which scales as M). For each training example, we maximize this lower bound with respect to the variational parameters $\boldsymbol{\mu}$ for fixed parameters $\boldsymbol{\theta}$, which results in the mean-field fixed-point equations:

$$\mu_j^{(1)} \leftarrow \sigma\left(\sum_{k=1}^{K} W_{jk}\left(\hat{v}_k + M\mu_k^{(2)}\right)\right), \qquad (9)$$

$$\mu_k^{(2)} \leftarrow \frac{\exp\left(\sum_{j=1}^{F} W_{jk}\mu_j^{(1)}\right)}{\sum_{k'=1}^{K}\exp\left(\sum_{j=1}^{F} W_{jk'}\mu_j^{(1)}\right)}, \qquad (10)$$

where $\sigma(x) = 1/(1+\exp(-x))$ is the logistic function. To solve these fixed-point equations, we simply cycle through layers, updating the mean-field parameters within a single layer.

Given the variational parameters $\boldsymbol{\mu}$, the model parameters $\boldsymbol{\theta}$ are then updated to maximize the variational bound using an MCMC-based stochastic approximation [10, 15, 18]. Let $\boldsymbol{\theta}_t$ and $\mathbf{x}_t = \{\mathbf{V}_t, \mathbf{h}^{(1)}{}_t, \mathbf{h}^{(2)}{}_t\}$ be the current parameters and the state. Then $\mathbf{x}_t$ and $\boldsymbol{\theta}_t$ are updated sequentially as follows: given $\mathbf{x}_t$, sample a new state $\mathbf{x}_{t+1}$ using alternating Gibbs sampling. A new parameter $\boldsymbol{\theta}_{t+1}$ is then obtained by making a gradient step, where the intractable model's expectation $\mathbb{E}_{P_{\text{model}}}[\cdot]$ in the gradient is replaced by a point estimate at sample $\mathbf{x}_{t+1}$.

In practice, to deal with variable document lengths, we take a minibatch of data and run one Markov chain for each training case for a few steps. To update the model parameters, we use an average over those chains. Similar to Contrastive Divergence learning, in order to provide a good starting point for the sampling, we initialize each chain at $\hat{\mathbf{h}}^{(1)}$ by sampling from the mean-field approximation to the posterior $q(\mathbf{h}^{(1)}|\mathbf{V})$.

3.2 An Efficient Pretraining Algorithm

The proper training procedure for the DBM model described above is quite slow. This makes it very impor-

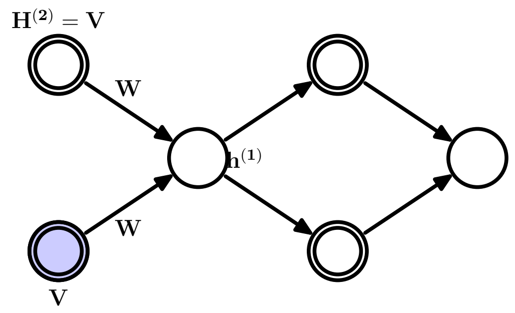


Figure 3: Pretraining a two-layer Boltzmann Machine using one-step contrastive divergence. The second hidden softmax layer is initialized to be the same as the observed data. The units in the first hidden layer have stochastic binary states, but the reconstructions of both the visible and second hidden layer use probabilities, so both reconstructions are identical.

tant to pretrain the model so that the model parameters start off in a nice region of space. Fortunately, due to parameter sharing between the visible and hidden softmax units, there exists an efficient pretraining method which makes the proper training almost redundant.

Consider a DBM with N observed and M hidden softmax units. Let us first assume that the number of hidden softmaxes M is the same as the number of words N in a given document. If we were given the initial state vector $\mathbf{H^{(2)}}$, we could train this DBM using one-step contrastive divergence with mean-field reconstructions of both the states of the visible and the hidden softmax units, as shown in Fig. 3. Since we are not given the initial state, one option is to set $\mathbf{H^{(2)}}$ to be equal to the data $\mathbf{V}$. Provided we use mean-field reconstructions for both the visible and second-layer hidden units, one-step contrastive divergence is then exactly the same as training a Replicated Softmax RBM with only one hidden layer but with bottom-up weights that are twice the top-down weights.

To pretrain a DBM with different number of visible and hidden softmaxes, we train an RBM with the bottom-up weights scaled by a factor of $1 + \frac{M}{N}$. In other words, in place of using $\mathbf{W}$ to compute the conditional probability of the hidden units (see Eq. 3), we use $(1 + \frac{M}{N})\mathbf{W}$:

$$P(h_j^{(1)} = 1|\mathbf{V}) = \sigma\left((1 + \frac{M}{N})\sum_{k=1}^{K} v_k W_{kj}\right). \qquad (11)$$

The conditional probability of the observed softmax units remains the same as in Eq. 4. This procedure is equivalent to training an RBM with $N + M$ observed visible units with each of the M extra units set to be the empirical word distribution in the document, i.e.. for $i \in \{N + 1, \ldots, N + M\}$,

$$v_{ik} = \frac{\sum_{j=1}^{N} v_{jk}}{\sum_{j=1}^{N}\sum_{k'=1}^{K} v_{jk'}}$$

Thus the M extra units are not 1-of-K, but represent distributions over the K words[3].

This way of pretraining the Over-Replicated Softmax DBMs with tied weights will not in general maximize the likelihood of the weights. However, in practice it produces models that reconstruct the training data well and serve as a good starting point for generative fine-tuning of the two-layer model.

3.3 Inference

The posterior distribution $P(\mathbf{h}^{(1)}|\mathbf{V})$ represents the latent topic structure of the observed document. Conditioned on the document, these activation probabilities can be inferred using the mean-field approximation used to infer data-dependent statistics during training.

A fast alternative to the mean-field posterior is to multiply the visible to hidden weights by a factor of $1 + \frac{M}{N}$ and approximate the true posterior with a single matrix multiply, using Eq. 11. Setting $M = 0$ recovers the proper posterior inference step for the standard Replicated Softmax model. This simple scaling operation leads to significant improvements. The results reported for retrieval and classification experiments used the fast pretraining and fast inference methods.

3.4 Choosing M

The number of hidden softmaxes M affects the strength of the additional prior. The value of M can be chosen using a validation set. Since the value of M is fixed for all Over-Replicated DBMs, the effect of the prior will be less for documents containing many words. This is particularly easy to see in Eq. 11. As N becomes large, the scaling factor approaches 1, diminishing the part of implicit prior coming from the M hidden softmax units. Thus the value of M can be chosen based on the distribution of lengths of documents in the corpus.

4 Experiments

In this section, we evaluate the Over-Replicated Softmax model both as a generative model and as a feature extraction method for retrieval and classification. Two datasets are used - 20 Newsgroups and Reuters Corpus Volume I (RCV1-v2).

[3]Note that when $M = N$, we recover the setting of having the bottom-up weights being twice the top-down weights.

4.1 Description of datasets

The 20 Newsgroups dataset consists of 18,845 posts taken from the Usenet newsgroup collection. Each post belongs to exactly one newsgroup. Following the preprocessing in [12] and [7], the data was partitioned chronologically into 11,314 training and 7,531 test articles. After removing stopwords and stemming, the 2000 most frequent words in the training set were used to represent the documents.

The Reuters RCV1-v2 contains 804,414 newswire articles. There are 103 topics which form a tree hierarchy. Thus documents typically have multiple labels. The data was randomly split into 794,414 training and 10,000 test cases. The available data was already preprocessed by removing common stopwords and stemming. We use a vocabulary of the 10,000 most frequent words in the training dataset.

4.2 Training details

The Over-Replicated Softmax model was first pretrained with Contrastive Divergence using the weight scaling technique described in Sec. 3.2. Minibatches of size 128 were used. A validation set was held out from the training set for hyperparameter selection (1,000 cases for 20 newsgroups and 10,000 for RCV1-v2). The value of M and number of hidden units were chosen over a coarse grid using the validation set. Typically, $M = 100$ performed well on both datasets. Increasing the number of hidden units lead to better performance on retrieval and classification tasks, until serious overfitting became a problem around 1000 hidden units. For perplexity, 128 hidden units worked quite well and having too many units made the estimates of the partition function obtained using AIS unstable. Starting with CD-1, the number of Gibbs steps was stepped up by one after every 10,000 weight updates till CD-20. Weight decay was used to prevent overfitting. Additionally, in order to encourage sparsity in the hidden units, KL-sparsity regularization was used. We decayed the learning rate as $\frac{\epsilon_0}{1+t/T}$, with $T = 10,000$ updates. This approximate training was sufficient to give good results on retrieval and classification tasks. However, to obtain good perplexity results, the model was trained properly using the method described in Sec. 3.1. Using 5 steps for mean-field inference and 20 for Gibbs sampling was found to be sufficient. This additional training gave improvements in terms of perplexity but the improvement on classification and retrieval tasks was not statistically significant.

We also implemented the standard Replicated Softmax model. The training procedure was the same as the pretraining process for the Over-Replicated Softmax model. Both the models were implemented on GPUs. Pretraining took 3-4 hours for the 2-layered Boltzmann Machines (depending on M) and the proper training took 10-12 hours. The DocNADE model was run using the publicly available code[4]. We used default settings for all hyperparameters, except the learning rates which were tuned separately for each hidden layer size and data set.

Table 1: Comparison of the average test perplexity per word. All models use 128 topics.

	20 News	**Reuters**
Training set size	11,072	794,414
Test set size	7,052	10,000
Vocabulary size	2,000	10,000
Avg Document Length	51.8	94.6
Perplexities		
Unigram	1335	2208
Replicated Softmax	965	1081
Over-Rep. Softmax ($M = 50$)	961	1076
Over-Rep. Softmax ($M = 100$)	**958**	**1060**

4.3 Perplexity

We compare the Over-Replicated Softmax model with the Replicated Softmax model in terms of perplexity. Computing perplexities involves computing the partition functions for these models. We used Annealed Importance Sampling [9] for doing this. In order to get reliable estimates, we ran 128 Markov chains for each document length. The average test perplexity per word was computed as $\exp\left(-1/L \sum_{l=1}^{L} 1/N_l \log p(\mathbf{v}_l)\right)$, where N_l is the number of words in document l. Table 1 shows the perplexity averaged over $L = 1000$ randomly chosen test cases for each data set. Each of the models has 128 latent topics. Table 1 shows that the Over-Replicated Softmax model assigns slightly lower perplexity to the test data compared to the Replicated Softmax model. For the Reuters data set the perplexity decreases from 1081 to 1060, and for 20 Newsgroups, it decreases from 965 to 958. Though the decrease is small, it is statistically significant since the standard deviation was typically ± 2 over 10 random choices of 1000 test cases. Increasing the value of M increases the strength of the prior, which leads to further improvements in perplexities. Note that the estimate of the log probability for 2-layered Boltzmann Machines is a lower bound on the actual log probability. So the perplexities we show are upper bounds and the actual perplexities may be lower (provided the estimate of the partition function is close to the actual value).

[4] `http://www.dmi.usherb.ca/~larocheh/code/DocNADE.zip`

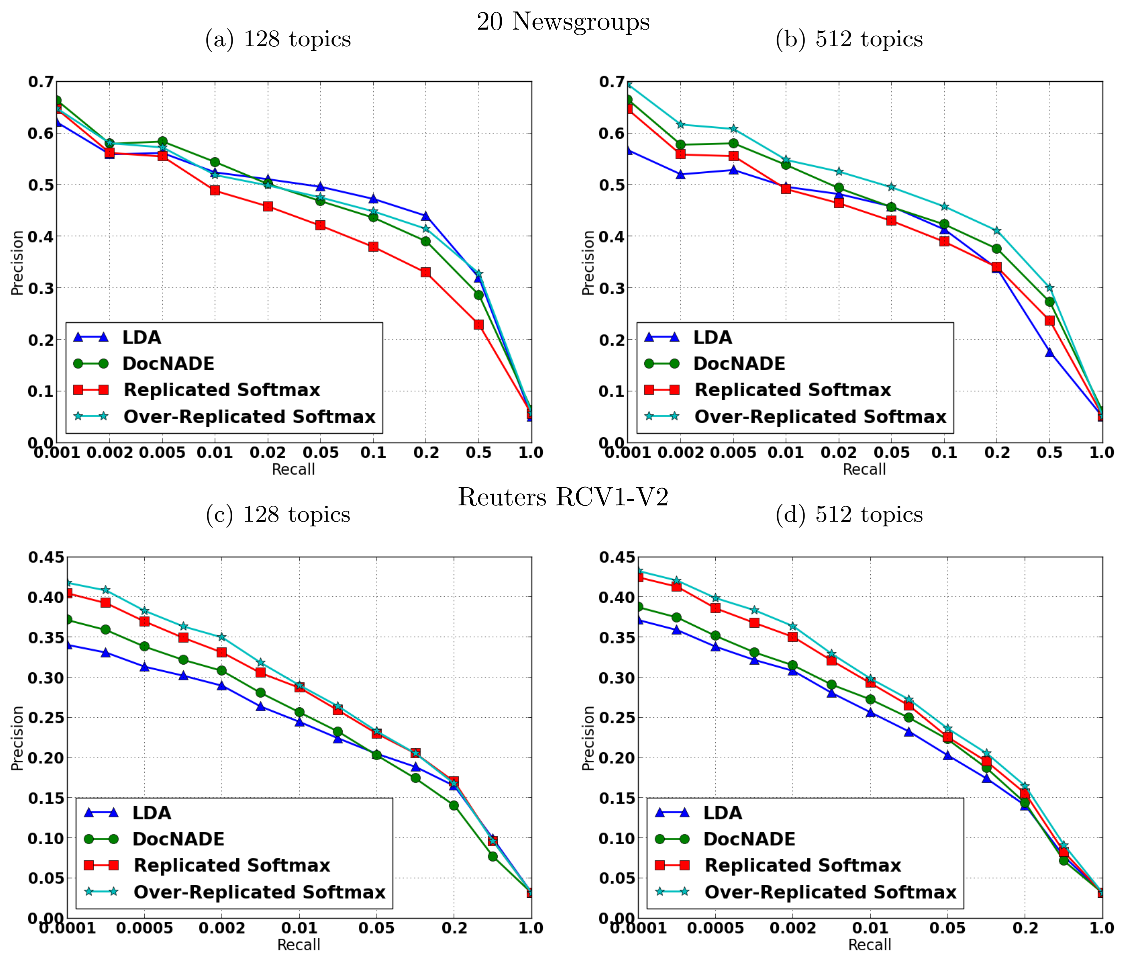


Figure 4: Comparison of Precision-Recall curves for document retrieval. All Over-Replicated Softmax models use $M = 100$ latent words.

4.4 Document Retrieval

In order to do retrieval, we represent each document $\mathbf{V}$ as the conditional posterior distribution $P(\mathbf{h}^{(1)}|\mathbf{V})$. This can be done exactly for the Replicated Softmax and DocNADE models. For two-layered Boltzmann Machines, we extract this representation using the fast approximate inference as described in Sec. 3.3. Performing more accurate inference using the mean-field approximation method did not lead to statistically different results. For the LDA, we used 1000 Gibbs sweeps per test document in order to get an approximate posterior over the topics.

Documents in the training set (including the validation set) were used as a database. The test set was used as queries. For each query, documents in the database were ranked using cosine distance as the similarity metric. The retrieval task was performed separately for each label and the results were averaged. Fig. 4 compares the precision-recall curves. As shown by Fig. 4, the Over-Replicated Softmax DBM outperforms other models on both datasets, particularly when retrieving the top few documents.

To find the source of improvement, we analyzed the effect of document length of retrieval performance. Fig. 5 plots the average precision obtained for query documents arranged in order of increasing length. We found that the Over-Replicated Softmax model gives large gains on documents with small numbers of words, confirming that the implicit prior imposed using a fixed value of M has a stronger effect on short documents. As shown in Fig. 5, DocNADE and Replicated Softmax models often do not do well for documents with few words. On the other hand, the Over-Replicated softmax model performs significantly better for short documents. In most document collections, the length of documents obeys a power law distribution. For example, in the 20 newsgroups dataset 50% of the documents have fewer than 35 words (Fig. 5c). This makes it very important to do well on short documents. The Over-Replicated Softmax model achieves this goal.

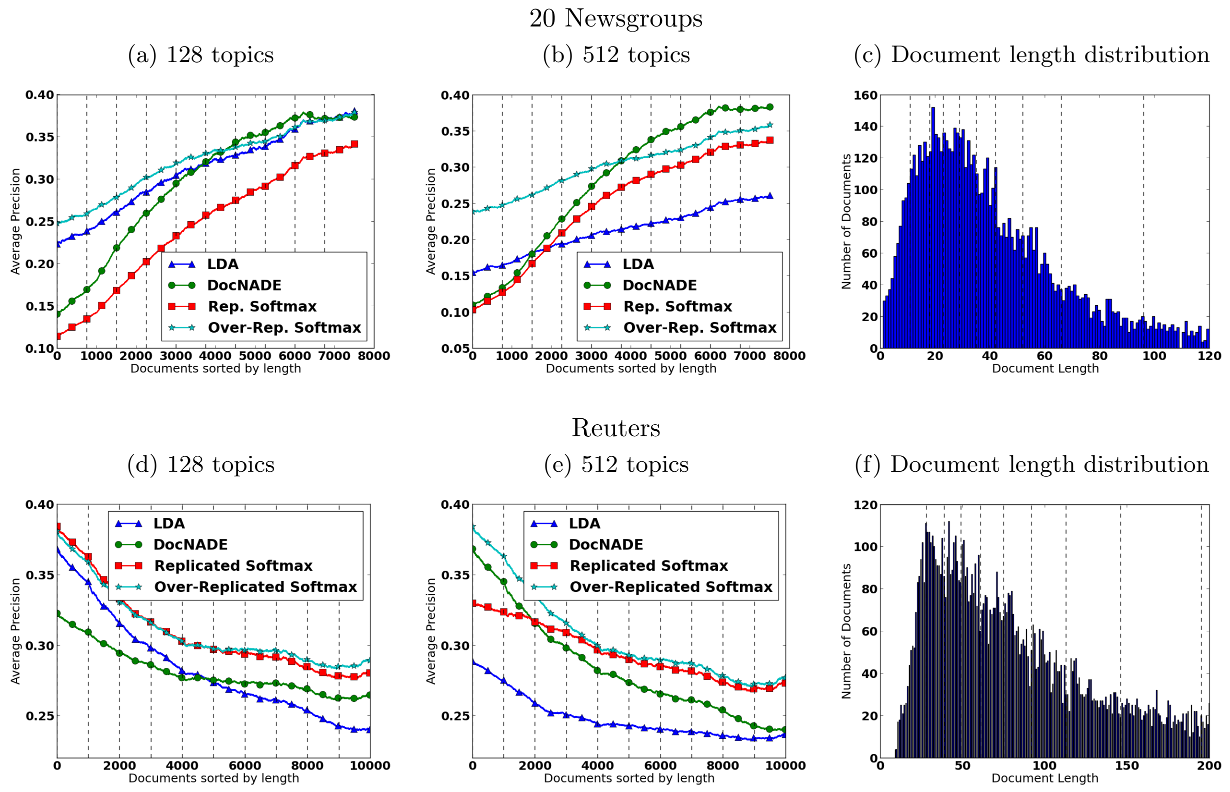


Figure 5: Effect of document size on retrieval performance for different topic models. The x-axis in Figures (a), (b), (d), (e) represents test documents arranged in increasing order of their length. The y-axis shows the average precision obtained by querying that document. The plots were smoothed to make the general trend visible. Figures (c) and (f) show the histogram of document lengths for the respective datasets. The dashed vertical lines denote 10-percentile boundaries. **Top :** Average Precision on the 20 Newsgroups dataset. **Bottom :** Mean Average Precision on the Reuters dataset. The Over-Replicated Softmax models performs significantly better for documents with few words. The adjoining histograms in each row show that such documents occur quite frequently in both data sets.

4.5 Document Classification

In this set of experiments, we evaluate the learned representations from the Over-Replicated Softmax model for the purpose of document classification. Since the objective is to evaluate the quality of the representation, simple linear classifiers were used. Multinomial logistic regression with a cross entropy loss function was used for the 20 newsgroups data set. The evaluation metric was classification accuracy. For the Reuters dataset, we used independent logistic regressions for each label since it is a multi-label classification problem. The evaluation metric was Mean Average Precision.

Table 2 shows the results of these experiments. The Over-Replicated Softmax model performs significantly better than the standard Replicated Softmax model and LDA across different network sizes on both datasets. For the 20 newsgroups dataset using 512 topics, LDA gets 64.2% accuracy. Replicated Softmax (67.7%) and DocNADE (68.4%) improve upon this. The Over-Replicated Softmax model further improves the result to 69.4%. The difference is larger for the Reuters dataset. In terms of Mean Average Precision (MAP), the Over-Replicated Softmax model achieves 0.453 which is a very significant improvement upon DocNADE (0.427) and Replicated Softmax (0.421).

Table 2: Comparison of Classification accuracy on 20 Newsgroups dataset and Mean Average Precision on Reuters RCV1-v2.

Model	**20 News**		**Reuters**	
	128	**512**	**128**	**512**
LDA	65.7	64.2	0.304	0.351
DocNADE	**67.0**	68.4	0.388	0.417
Replicated Softmax	65.9	67.7	0.390	0.421
Over-Rep. Softmax	66.8	**69.1**	**0.401**	**0.453**

We further examined the source of improvement by analyzing the effect of document length on the classification performance. Similar to retrieval, we found that the Over-Replicated Softmax model performs well on short documents. For long documents, the performance of the different models was similar.

5 Conclusion

The Over-Replicated Softmax model described in this paper is an effective way of defining a flexible prior over the latent topic features of an RBM. This model causes no increase in the number of trainable parameters and only a minor increase in training algorithm complexity. Deep Boltzmann Machines are typically slow to train. However, our fast approximate training method makes it possible to train the model with CD, just like an RBM. The features extracted from documents using the Over-Replicated Softmax model perform better than features from the standard Replicated Softmax and LDA models and are comparable to DocNADE across different network sizes.

While the number of hidden softmax units M, controlling the strength of the prior, was chosen once and fixed across all DBMs, it is possible to have M depend on N. One option is to set $M = cN$, $c > 0$. In this case, for documents of all lengths, the second-layer would perform perform $^{c}/_{c+1}$ of the modeling work compared to the first layer. Another alternative is to set $M = N_{max} - N$, where N_{max} is the maximum allowed length of all documents. In this case, our DBM model will always have the same number of replicated softmax units $N_{max} = N + M$, hence the same architecture and a single partition function. Given a document of length N, the remaining $N_{max} - N$ words can be treated as missing. All of these variations improve upon the standard Replicated Softmax model, LDA, and DocNADE models, opening up the space of new deep undirected topics to explore.

References

[1] David M. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, 2012.

[2] David M. Blei, Thomas L. Griffiths, and Michael I. Jordan. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *J. ACM*, 57(2), 2010.

[3] David M. Blei, Andrew Ng, and Michael Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[4] K. Canini, L. Shi, and T. Griffiths. Online inference of topics with latent Dirichlet allocation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 5, 2009.

[5] T. Griffiths and M. Steyvers. Finding scientific topics. In *Proceedings of the National Academy of Sciences*, volume 101, pages 5228–5235, 2004.

[6] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1711–1800, 2002.

[7] Hugo Larochelle and Stanislas Lauly. A neural autoregressive topic model. In *Advances in Neural Information Processing Systems 25*, pages 2717–2725. 2012.

[8] D. Mimno and A. McCallum. Topic models conditioned on arbitrary features with dirichlet-multinomial regression. In *UAI*, pages 411–418, 2008.

[9] Radford M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, April 2001.

[10] R. R. Salakhutdinov and G. E. Hinton. Deep Boltzmann machines. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 12, 2009.

[11] Ruslan Salakhutdinov and Geoff Hinton. A better way to pretrain deep boltzmann machines. In *Advances in Neural Information Processing Systems 25*, pages 2456–2464. 2012.

[12] Ruslan Salakhutdinov and Geoffrey Hinton. Replicated softmax: an undirected topic model. In *Advances in Neural Information Processing Systems 22*, pages 1607–1614. 2009.

[13] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[14] Y. W. Teh, K. Kurihara, and M. Welling. Collapsed variational inference for HDP. In *Advances in Neural Information Processing Systems*, volume 20, 2008.

[15] T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *ICML*. ACM, 2008.

[16] Chong Wang and David M. Blei. Variational inference for the nested chinese restaurant process. In *NIPS*, pages 1990–1998, 2009.

[17] Eric P. Xing, Rong Yan, and Alexander G. Hauptmann. Mining associated text and images with dual-wing harmoniums. In *UAI*, pages 633–641. AUAI Press, 2005.

[18] L. Younes. On the convergence of Markovian stochastic algorithms with rapidly decreasing ergodicity rates, March 17 2000.

Speedy Model Selection (SMS) for Copula Models

Yaniv Tenzer and Gal Elidan
Department of Statistics, The Hebrew University

Abstract

We tackle the challenge of efficiently learning the structure of expressive multivariate real-valued densities of copula graphical models. We start by theoretically substantiating the conjecture that for many copula families the magnitude of Spearman's rank correlation coefficient is monotonic in the expected contribution of an edge in network, namely the negative copula entropy. We then build on this theory and suggest a novel Bayesian approach that makes use of a prior over values of Spearman's rho for learning copula-based models that involve a mix of copula families. We demonstrate the generalization effectiveness of our highly efficient approach on sizable and varied real-life datasets.

1 Introduction

Learning expressive real-valued multivariate distributions is of central interest in numerous fields ranging from computational biology to economics to climatology. When the joint distribution of interest is far from multivariate normal, this modeling task can be a great challenge. In statistics, copulas [Joe, 1997, Nelsen, 2007] are the central tool for capturing flexible multivariate real-valued distributions by separating the choice of the univariate marginals and the copula function that links them. Copulas are typically only effective in low dimensions, and much of the research in the field in fact focuses on the bivariate case. Accordingly, in the last decade, various high-dimensional constructions that build on a collection of copulas have been suggested, most notably those based on the vine construction [Bedford and Cooke, 2002, Kurowicka and Cooke, 2002]. These models have proved to be quite effective for crossing the *few variable* barrier. However, in the context of many tens of variables to hundreds and thousands of variables, applications have been few and involve costly and time-consuming expert elicitation [Hanea et al., 2010].

In machine learning, probabilistic graphical models, and in particular directed Bayesian networks (BNs) [Pearl, 1988], have become increasingly popular as a flexible and intuitive framework for modeling multivariate densities based on a qualitative graph structure $\mathcal{G}$ that encodes the independencies in the domain. Graphical models are geared toward the high-dimensional case and numerous algorithms for estimation, model selection and prediction using these models have been developed in recent decades [Koller and Friedman, 2009]. Unfortunately, due to computational considerations, real-valued high-dimensional modeling using this framework is often limited to a structured multivariate Gaussian model.

In recent years, several works suggested a fusion between copulas and graphical models [Kirshner, 2007, Elidan, 2010], with the goal of allowing for flexible real-valued modeling that is practical in the high-dimensional setting. The basic idea is that the joint density is defined via a collection of local copula functions that capture the direct dependence between a variable and its parents in the graph $\mathcal{G}$, as well as a set of univariate marginals that are shared across the entire model. As with standard graphical models, the super-exponential task of learning the structure of such models from data poses practical difficulties. Specifically, the computational bottleneck of structure learning is the assessment of the quality of candidate structures, which in turn requires costly estimation of maximum likelihood parameters. Indeed, even when using a simple greedy procedure to traverse the space of structures, or when limiting ourselves to tree structured models, structure learning can be computationally demanding for non-Gaussian models. Our goal in this work is to cope with this challenge.

Recently, Elidan [2012] suggested a highly efficient approach for learning the structure of copula-based

Bayesian networks. Briefly, the building block of structure learning is the *ranking* of the merit of an edge $X \to Y$ given M training samples. Using $U \equiv F_X(x)$, $V \equiv F_Y(y)$ to denote the marginal distributions of these variables and assuming a copula-based model, they note that the benefit of the edge is asymptotically equal to the negative differential entropy

$$-H(c_\theta(U,V)) = \int c_\theta(u,v) \log c_\theta(u,v) du dv, \qquad (1)$$

where $c(\cdot)$ denotes the (copula) density that corresponds to the joint distribution of X and Y. They then suggest that $-H(c_\theta(U,V))$ is monotonic in the easy to compute Spearman's ρ_s measure of correlation. This in turn facilitates highly efficient structure learning where simple Spearman's ρ_s computations are used to rank candidate edge modifications. The monotonicity is proved for the Gaussian copula and algebraically simple Farlie-Gumbel-Morgenstern (FGM) copula family. Based on simulations, they further conjectured that the result holds for several additional copula families. Finally, they show that the method can be used to learn the structure of copula-based models that generalize well very efficiently. The method's main limitation, other than the gap in theory, is the fact that the same copula family is used to parameterize all edges in the model.

In this work we extend Elidan [2012] along two important axes. First, we provide a formal proof of the monotonicity conjecture given a sufficient condition that applies to a wide range of common copula families, a novel contribution to the theory of copulas on its own. Second, we tackle the challenge of performing structure learning while also allowing for a mixed combination of copulas, thereby significantly increasing the expressive power of the model. Briefly, our theoretical result suggests that the selection between copula families can be made based on expected likelihood *characteristic curves* that are computed *once.* A natural Bayesian prior is then used to "calibrate" the curves for several families. Finally, the posterior curves are used to select a copula family for each edge based *only* on Spearman's ρ_s computations.

We use our speedy model selection (SMS) approach to learn copula-based tree structured models for several real-life datasets that are quite substantial in size in the context of structure learning with the number of variables ranging from 100 to close to 900. In all cases, we demonstrate impressive performance benefits relative to learning a model that is constrained to using a single copula family. Further, in many instances we show that our highly efficient approach is competitive with the computationally demanding golden standard where the best copula for each edge is computed via *costly* maximum likelihood estimation for each family.

2 Background

In this section we briefly provide the necessary background on copulas, Spearman's ρ_s and stochastic orders of multivariate distributions.

2.1 Copula and Spearman's ρ_s

A copula function joins univariate marginals into a joint real-valued multivariate distribution. Formally,

Definition 2.1: Let $U_1, \dots, U_n$ be random variables marginally uniformly distributed on $[0,1]$. A copula function $C : [0,1]^n \to [0,1]$ 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 function. ∎

Now consider an arbitrary set $\mathcal{X} = \{X_1, \dots X_n\}$ of real-valued random variables (typically *not* marginally uniformly distributed). Sklar's seminal theorem [Sklar, 1959] states that for *any* joint distribution $F_{\mathcal{X}}(\mathbf{x})$, there exists a copula function C such that

$$F_{\mathcal{X}}(\mathbf{x}) = C(F_1(x_1), \dots, F_n(x_n)).$$

When the univariate marginals are continuous, C is uniquely defined.

The constructive converse, which is of central interest from a modeling perspective, is also true. Since $U_i \equiv F_i$ is itself a random variable that is always uniformly distributed in $[0,1]$, *any* copula function taking *any* marginal distributions $\{F_i(x_i)\}$ as its arguments, defines a valid joint 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* $f(\mathbf{x}) = \frac{\partial^n F(x_1,\dots,x_n)}{\partial x_1 \dots \partial x_n}$ 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 $c(F_1(x_1), \dots, F_n(x_n))$, is called the *copula density function.*

Example 2.2: The Gaussian copula is undoubtedly the most commonly used copula family and is defined as

$$C_\Sigma(\{U_i\}) = \Phi_\Sigma\left(\Phi^{-1}(U_1), \dots, \Phi^{-1}(U_N)\right), \qquad (2)$$

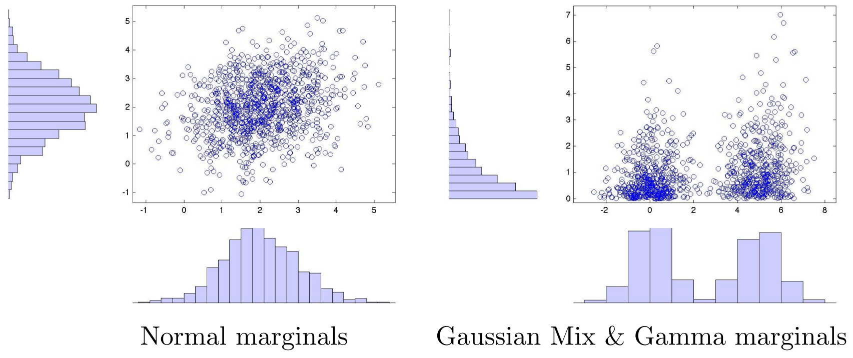


Figure 1: Samples from the bivariate Gaussian copula with correlation $\theta = 0.25$. (left) with unit variance Gaussian marginals; (right) with a mixture of Gaussian and Gamma marginals.

where Σ is a correlation matrix, Φ is the standard normal distribution, and Φ_Σ is a zero mean normal distribution with correlation matrix Σ. Figure 1 exemplifies the flexibility that comes with this seemingly limited elliptical copula family. ∎

Copula are intimately connected to many dependence concepts such as Spearman's ρ_s measure of association

$$\rho_s(X_1, X_2) = \frac{cov(F_{X_1}, F_{X_2})}{STD(X_1)STD(X_2)},$$

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

$$\rho_s(X_1, X_2) = \rho_s(C) \equiv 12 \int\int C(u, v)dudv - 3.$$

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.

2.2 Stochastic Orderings

The vast majority of copula families are parameterized by a dependence parameter θ that defines a stochastic ordering in the bivariate case. Below we define two such related orders that will be used in the sequel. In the rest of the paper, we use $\mathbf{X}$, $\mathbf{Y}$ to denote bivariate random vectors with distributions $F_{\mathbf{X}}(u, v)$ and $F_{\mathbf{Y}}(u, v)$, respectively.

Definition 2.3: $\mathbf{Y}$ is said to be more positive quadrant dependent than $\mathbf{X}$, denoted by $\mathbf{X} \leq_{PQD} \mathbf{Y}$, if $\forall(u, v) \in \mathbf{R}^2, F_{\mathbf{X}}(u, v) \leq F_{\mathbf{Y}}(u, v)$. ∎

Thus, PQD ordering corresponds to a fast accumulation of density. To define the second ordering, we first need the notion of supermodularity. In what follows we use the following notation: $u \vee v \equiv min(u, v)$, $u \wedge v \equiv max(u, v)$.

Definition 2.4: A function $\Psi : \mathbf{R}^2 \rightarrow \mathbf{R}$ is said to be *supermodular* if

$$\forall(u, v) \in \mathbf{R}^2, \Psi(u \vee v) + \Psi(u \wedge v) \geq \Psi(u) + \Psi(v)$$

Ψ is submodular when the inequality is reversed. ∎

The supermodular ordering can now be defined:

Definition 2.5: $\mathbf{Y}$ is said to be greater than $\mathbf{X}$ in the supermodular order, denoted by $\mathbf{X} \leq_{sm} \mathbf{Y}$, if $\forall\Psi$ such that Ψ is super modular: $E[\Psi(\mathbf{X})] \leq E[\Psi(\mathbf{Y})]$ ∎

This property is important in our context since, in the bivariate case, we have the following result due to Shaked and Shanthikumar [2007]:

Theorem 2.6: $\mathbf{X} \leq_{PQD} \mathbf{Y} \iff \mathbf{X} \leq_{sm} \mathbf{Y}$.

3 Monotonicity of the Copula Entropy in the Dependence Parameter

As discussed in the introduction, Elidan [2012] suggested that the magnitude of Spearman's ρ_s is monotonic in the negative copula entropy which in turn asymptotically approximates the expected log-likelihood of a model, thereby giving rise to an efficient structure learning procedure. In this section we prove the conjecture for a wide range of copula families and discuss its relationship to real-valued majorization. In the next section we present a novel algorithmic approach called speedy model selection (SMS) that builds on this theory and allows us to efficiently perform structure learning while *at the same time* choose the local copula family.

3.1 TP2 Density Implies Entropy Ordering

We now present our central result, namely the identification of a widely applicable sufficient condition for the monotonicity of the copula entropy in the dependence parameter, and consequently in Spearman's ρ_s.

Recall that $\mathbf{X}$, $\mathbf{Y}$ are two bivariate random vectors. Throughout this section let $\mathbf{X} \sim C_{\theta_1}(u, v), \mathbf{Y} \sim$

$C_{\theta_2}(u,v)$ with $\theta_1 < \theta_2$, where $C_\theta(u,v)$ is an absolutely continuous bivariate copula family that is increasing in $<_{PQD}$ so that the cumulative distribution of $\mathbf{Y}$ is greater than that of $\mathbf{X}$ for all (u,v). Note that essentially all copula families that are parameterized by a so called dependence parameter θ are PQD ordered.

Before stating the main result, using $u \vee v \equiv min(u,v)$, $u \wedge v \equiv max(u,v)$, we define the following notion:

Definition 3.1: A function $\Psi : \mathbf{R}^2 \to \mathbf{R}$ is TP2 (total positive of order 2) if the following holds:

$$\forall (u,v) \in \mathbf{R}^2 \quad \Psi(u \vee v) \cdot \Psi(u \wedge v) \geq \Psi(u) \cdot \Psi(v).$$

Ψ is called RR2 (reversed regular of order 2) when the inequality is reversed. ∎

Note that the density for many copulas is a TP2 or RR2 function (for example, 8 of the twelve B1-B12 families defined in Joe [1997] are known to be TP2 and the property may hold for some of the others).

The following property of TP2 (RR2) functions, easily proved using logarithmic properties, will be needed:

Observation 3.2: Given a positive function $\Psi(u,v)$ which is TP2 (RR2), $\Phi(u,v) = log(\Psi(u,v))$ is supermodular (submodular).

We are now ready for our central result:

Theorem 3.3: If $C_\theta(u,v)$ is a copula family that defines a positive PQD ordering, and the copula density $c_\theta(u,v)$ is TP2 for all values of θ, then

$$\theta_1 < \theta_2 \Rightarrow -H(c_{\theta_1}) \leq -H(c_{\theta_2})$$

When $C_\theta(u,v)$ is RR2 the inequality is reversed.

Proof: Recall that $\mathbf{X} \sim c_{\theta_1}$ and $\mathbf{Y} \sim c_{\theta_2}$, and that $\theta_1 < \theta_2$. We will show that the following holds:

$$\begin{aligned} -H(\mathbf{X}) &= \int c_{\theta_1}(u,v) log(c_{\theta_1}(u,v)) \mathrm{d}u\mathrm{d}v \\ &\leq \int c_{\theta_2}(u,v) log(c_{\theta_1}(u,v)) \mathrm{d}u\mathrm{d}v \\ &\leq \int c_{\theta_2}(u,v) log(c_{\theta_2}(u,v)) \mathrm{d}u\mathrm{d}v = -H(\mathbf{Y}) \end{aligned}$$

First inequality. Since $C_\theta(u,v)$ defines a PQD ordering, we have from Theorem 2.6 that $\mathbf{X} \leq_{sm} \mathbf{Y}$. Let $\Psi(u,v) = log(c_{\theta_1}(u,v))$. Since $c_{\theta_1}(u,v)$ is TP2, from Observation 3.2 we have that $\Psi(u,v)$ is super modular, that is $E(\Psi(\mathbf{X})) \leq E(\Psi(\mathbf{Y}))$. Thus:

$$\int c_{\theta_1}(u,v)\Psi(u,v)\mathrm{d}u\mathrm{d}v \leq \int c_{\theta_2}(u,v)\Psi(u,v)\mathrm{d}u\mathrm{d}v.$$

The first inequality follows by substitution of Ψ.

Family	CDF	condition
Normal	$\Phi_\theta(\Phi^{-1}(u), \Phi^{-1}(v))$	$0 \leq \theta \leq 1$
FGM	$uv + \theta uv(1-u)(1-v)$	$-1 \leq \theta \leq 1$
Gumbel	$e^{-[(\hat{u})^\theta + (\hat{v})^\theta)]^{1/\theta}}$	$1 \leq \theta \leq \infty$, $\hat{u} = -log(u)$
Frank	$-\frac{1}{\theta} \log \left(1 - \frac{\tau(u)\tau(v)}{\tau(1)}\right)$	$0 \leq \theta \leq \infty$, $\tau(x) = 1 - e^{-\theta x}$
Clayton	$\max\left(u^{-\theta}+v^{-\theta}-1, 0\right)^{-\frac{1}{\theta}}$	$\theta \in [-1, \infty], \neq 0$
Joe	$1 - \left(\bar{u}^\delta + \bar{v}^\delta - \bar{u}^\delta \bar{v}^\delta\right)^{\frac{1}{\delta}}$	$\delta \in [1, \infty)$, $\bar{u} = 1 - u$
AMH*	$\frac{uv}{1-\theta(1-u)(1-v)}$	$-1 \leq \theta \leq 1$
GB*	$uve^{-\theta ln(u) ln(v)}$	$0 \leq \theta \leq 1$

Table 1: TP2/RR2 Copula families. '*' marks families for which, to the best of our knowledge, this property was not previously known.

Second inequality. The difference between the two sides of the second inequality is

$$\int c_{\theta_2}(u,v) \left[log(c_{\theta_1}(u,v)) - log(c_{\theta_2}(u,v))\right] \mathrm{d}u\mathrm{d}v$$

The result follows by noting that this is simply the Kullback-Leibler divergence between the two densities c_{θ_2} and c_{θ_1}, and the fact that this divergence is always non-negative [Cover and Thomas, 1991]. ∎

Note that the above theorem is stated for positively PQD ordered copula families. For negatively ordered families (e.g., Gumbel-Barnett) a reverse monotone relationship holds, as can be similarly proved.

The following is an immediate consequence of Theorem 3.3 and the known monotonicity of ρ_s in the dependence parameter θ for PQD ordered families:

Corollary 3.4 : *If the copula density $c_\theta(u,v)$ is TP2/RR2 for all θ, then the magnitude of Spearman's ρ_s is monotonic in the copula entropy.*

Note that, phrased in terms of the *magnitude* of ρ_s, the result also holds for PQD families such as the Gaussian copula that are TP2 for one side of the parameter values ($0 \leq \theta \leq 1$) and RR2 otherwise ($-1 \leq \theta \leq 0$).

3.2 Examples of TP2/RR2 Copulas

As discussed, the density of many copulas is a TP2/RR2 function making our theoretical result widely applicable. Table 1 lists these copula families and provides their distribution function.

The Gaussian, Fairlie-Gumbel-Morgenstern (FGM), Frank, Gumbel, Clayton and Joe copulas are all known to have a TP2/RR2 density [Joe, 1997]. Thus, for all these families the negative entropy is monotonic in θ, and consequently in the magnitude of Spearman's ρ_s.

Two additional popular copula families are confirmed to have a TP2/RR2 density:

Lemma 3.5: *The Ali-Mikhail-haq (AMH) copula has a TP2 density for nonnegative θ values and an RR2 density otherwise. The Gumbel-Barnett (GB) copula has an RR2 density.*

Proof of this result can be found in Appendix 7.

3.3 Other Copula Families

For completeness, we now discuss another sufficient condition for the monotonicity of the entropy in the dependence parameter and its relation to the TP2 condition. To the best of our knowledge, other than the work of Elidan [2012] that formulated the conjecture proved above, the only work that sheds theoretical light on the relationship between the copula dependence parameter θ and the entropy is that of Joe [1987]. The relevant details are summarized below.

If f, g are two n-dimensional densities, then f is said to be majorized by g, denoted by $f \preceq g$, iff $\int \Phi(f)dx \leq \int \Phi(g)dx$ for all convex functions Φ. In particular, since $\Phi(x) = xlog(x)$ is convex, then letting $\mathbf{X}, \mathbf{Y}$ be two n-dimensional random vectors, such that $\mathbf{X} \sim f_1$, $\mathbf{Y} \sim f_2$, and f_1 is majorized by f_2, we have that

$$\begin{aligned} -H(\mathbf{X}) &\equiv \int f_1 log(f_1) d\mathbf{X} \\ &\leq \int f_2 log(f_2) d\mathbf{Y} \equiv -H(\mathbf{Y}). \end{aligned}$$

With some additional technical details (see [Joe, 1987]), it is possible to show that for all elliptical copula families the dependence parameter θ implies a majorization ordering, which in turn implies monotonicity of the entropy in the absolute value of Spearman's ρ_s.

Thus, the monotonicity of the entropy in the correlation parameter for a bivariate Gaussian copula can be proved via majorization or Theorem 3.3. However, majorization *does not* hold for the other families which have a TP2 density. Conversely, the t-copula elliptical family defines a majorization ordering but its density is neither TP2 nor RR2 for some degrees of freedom [Allan.R.Sampson, 1983].

Finally, the widely used Plackett family of copulas has a density that is neither a TP2 function, nor does it define a majorization ordering. However, as the simulations of Elidan [2012] suggest, the monotonicity of the entropy in the dependence parameter also holds for this copula family. Identification of the conditions necessary for the monotonicity relationship to hold remains a future challenge.

4 A Bayesian Approach for Learning Expressive Copula Trees

Base on the theoretical developments presented in the previous section, we now present a speedy model selection (SMS) approach for learning the structure of copula-based graphical models while allowing for different copula families within the same model. For clarity and simplicity of exposition, we focus on the case of copula trees where the relationship between the theory and practice is most direct. As we shall see in Section 5, even in this seemingly simple setting, our approach offers significant generalization benefits. We start with a brief review of a copula tree model and how Spearman's ρ_s can be used to learn its structure for a single copula family.

4.1 Structure Learning using Spearman's ρ_s

We now we briefly review the idea put forth by Elidan [2012] for using Spearman's ρ_s to learn the structure of a copula network, an idea whose theoretical substantiation has been greatly increased by the developments in the previous section.

In a tree structured copula model [Kirshner, 2007, Elidan, 2010], the joint density is represented as a product of bivariate copula densities corresponding to the edges of the tree T and the univariate marginals:

$$f_{\mathcal{X}}(x_1, \ldots, x_n) = \prod_{(i,j) \in T} c_{ij}(F_i(x_i), F_j(x_k)) \prod_i f_i(x_i).$$

When learning the structure of a model, we seek a graph for which the (penalized) maximum likelihood function is highest. Since the likelihood function itself decomposes, the building block of learning is the evaluation of the merit of an edge $X \to Y$, independently of all other edges. In the case of the copula parameterization the relevant term is

$$Score(X, Y) \equiv \sum_{m=1}^{M} \log c_{\hat{\theta}}(F_X(x[m]), F_Y(y[m])),$$

where $\hat{\theta}$ are the estimated parameters, the sum is over training instances, and the marginal terms that do not depend on the graph's structure have been dropped.

Evaluation of $Score(X, Y)$ can be computationally difficult. However, all that we really need to identify the optimal tree is a *ranking* of the scores for all possible edges. If we assume that the data is generated from the copula, then as $M \to \infty$ we have that

$$Score(X, Y) \to -H(C_\theta(U, V)),$$

where U, V are the ranks of X, Y, respectively.

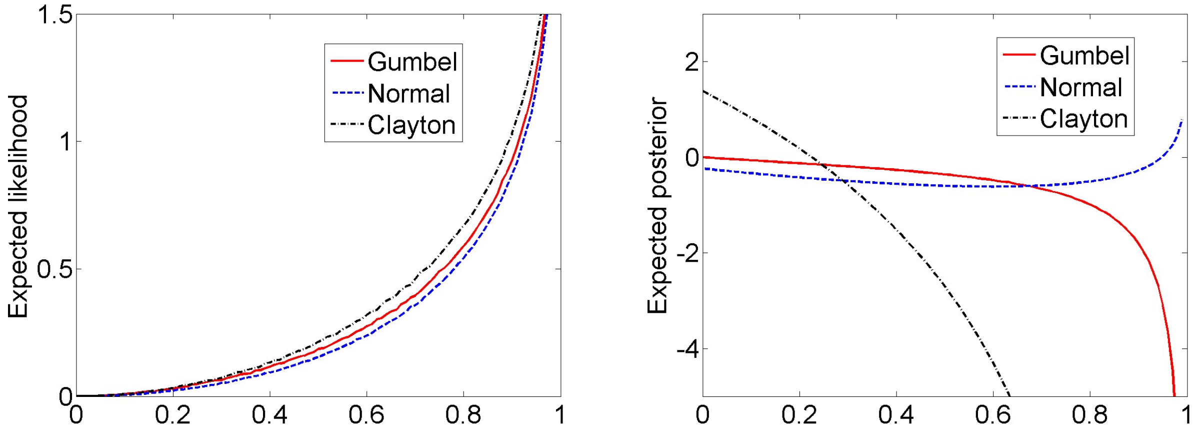


Figure 2: (left) expected log-likelihood vs. Spearman's ρ_s for the normal, Gumbel and Clayton copula families. (right) expected posterior vs. Spearman's ρ_s.

Thus, having proved that $|\rho_s|$ is monotonic in $-H(C_\theta(U,V))$, we can simply use an easy to compute empirical estimate of ρ_s to rank candidate edges, and find the optimal tree with respect to this measure using a simple maximum spanning tree algorithm.

4.2 Choosing From Multiple Copula Families

The above approach, suggested by Elidan [2012] allows for efficient structure learning of a copula based model that makes use of a *single* copula family. Obviously, not all pairs of variables share the same dependence characteristics and while the interaction between one pair of variable be be Gaussian, the interaction between another pair may be heavy-tailed and exhibit, for example, a behavior that is close to a Gumbel distribution. As an example, for the **Crime** census domain described in Section 5, the optimal tree includes around 41% edges parameterized by a Gaussian copula, 48% edges parameterized by a Gumbel copula, and 11% edges parameterized by a Clayton copula. Obviously, the need for a mix of copula families is real.

We now present an efficient approach for learning copula-based graphical models while allowing for a mix of copula families while retaining the lightning-speed efficiency of learning that is based on Spearman's ρ_s empirical evaluation. Naively, since the expected log-likelihood is monotonic in Spearman's ρ_s, the following procedure may seem reasonable:

- Simulate the characteristic curve of expected log-likelihood for each copula family (note that this needs to be carried out only once).
- For each value of Spearman's ρ_s choose the family that offers the highest expected log-likelihood

Unfortunately, such a procedure can fail since the theoretical result guarantees monotonicity *within* a copula family and not *between* copula families. In fact, as Figure 2(left) shows, the expected log-likelihood vs. Spearman's ρ_s is highest for Clayton copula family through much of the range of ρ_s values. Using the naive approach would lead us in this case to over-favor the Clayton copula.

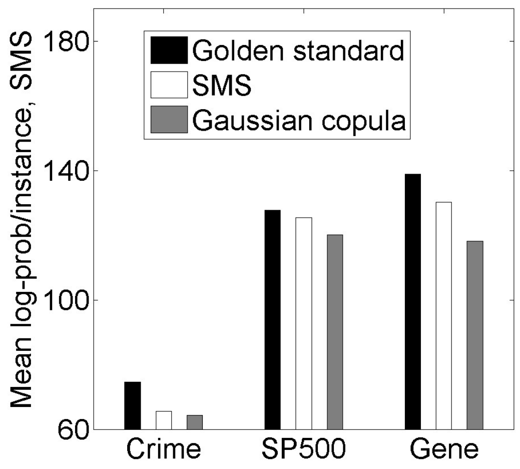
An intuitive explanation to the above phenomenon is that while characteristic curves indeed capture the behavior *given* Spearman's ρ_s for a particular family, they do not take into account the likelihood of seeing a particular value of ρ_s within each family. In fact, we can expect the density of ρ_s (which is always in the range $[-1,1]$) to be quite different between the normal copula family whose dependence parameter is in the range $[-1,1]$ and, for example, the Clayton copula family whose parameter has infinite support.

Given the above, we would like to somehow take into account a prior density over ρ_s for each copula family. Using $\mathcal{C}$ to denote the set of copula families and $f_c(\rho_s)$ to denote the density of ρ_s for a copula family $c \in \mathcal{C}$, we will then choose the copula family that maximize the expected posterior

$$argamax_{c\in\mathcal{C}} E\left(\log c(F(x),F(y);\rho_s) + \ logf_c(\rho_s)\right) \quad (3)$$

Importantly, this approach still relies on *precomputed* (posterior) characteristic curves and thus is as efficient as learning with a single copula family, regardless of the number of copula families considered.

The obvious question is how to choose the prior $f_c(\rho_s)$ for each copula family. In depth exploration of this question is left to future work and in here we use a straightforward approach which, as will be seen



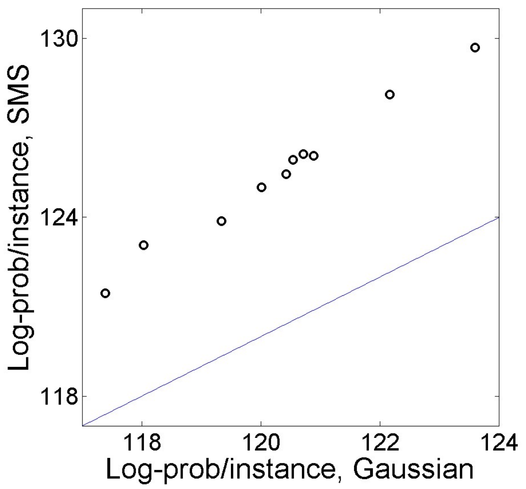


Figure 3: (left) Average log probability per instance over 10 folds. Compared are the Gaussian copula model, the model with a mix of copula families using our SMS method, and the golden standard of learning using exact maximum likelihood computations. Shown are results for the **Crime**, **SP500**, and **Gene** datasets. (right) comparison of our approach to the Gaussian copula baseline for all repetitions for the **SP500** domain.

in Section 5, proves quite effective in practice. We choose a prior that, while taking into account the range of dependence parameter for each copula family, assigns higher density to the independence model where $\rho_s = 0$. Appealingly, this is both uninformative while at the same time ensuring that we do not encourage dependence that is due to finite data noise.

Concretely, in this paper we consider the normal copula and the most popular Archimedean copula families, namely the Clayton, Frank, and Gumbel copula families. For the normal copula, the above translates into a standard truncated Laplace (also known as double exponential prior). For the Gumbel copula we use a shifted by unity exponential distribution (according to dependence parameter support) and for Clayton we take exponential distribution with parameter $\lambda = 4$. The resulting characteristic curves are shown in Figure 2(right) where it is clear that different copula families are preferred (highest) in different regions. In the next section we will show that using this curve to automatically choose the copula family based on empirical Spearman's ρ_s evaluation results in competitive models that are learned very efficiently.

We note that the characteristic curve for the Frank copula family is missing from this graph because of its similarity to that of the normal copula family. The implication of this similarity is that our method cannot be used to separate these two copula families. This should not come as a great surprise since the symmetric Frank density, while not Gaussian, is much more similar to the normal distribution than, for example, the asymmetric heavy-tailed Gumbel one. While this may sound problematic, due to the similarity of densities, the practical implications of wrongly choosing between the Frank and Gaussian copulas are relatively small, as confirmed in preliminary experiments (not shown here for clarity of exposition).

5 Experimental Evaluation

In this section we demonstrate the practical benefit of our speedy model selection (SMS) method for learning expressive real-valued copula graphical models. As noted, we focus on tree structures where the learning task decomposes into the bivariate evaluation of the merit of each edge in the network individually. The significant generalization advantage of copula-based graphical models over the standard Gaussian BN has been demonstrated in the past [Kirshner, 2007, Elidan, 2010] (and confirmed for our datasets). For clarity, in here we focus on the *additional* advantage over models that involve only a single copula family.

For each edge we allow for a Gaussian, Clayton or Gumbel copula, with the prior for each family as defined in Section 4. As a baseline we consider learning only with a Gaussian copula, which is the strongest of all single family baselines. We also compare to the golden standard of learning using exact maximum likelihood computations. For the univariate marginals in all cases, we use standard kernel-based approach [Parzen, 1962] with the common Gaussian kernel (see, for example, [Bowman and Azzalini, 1997] for details).

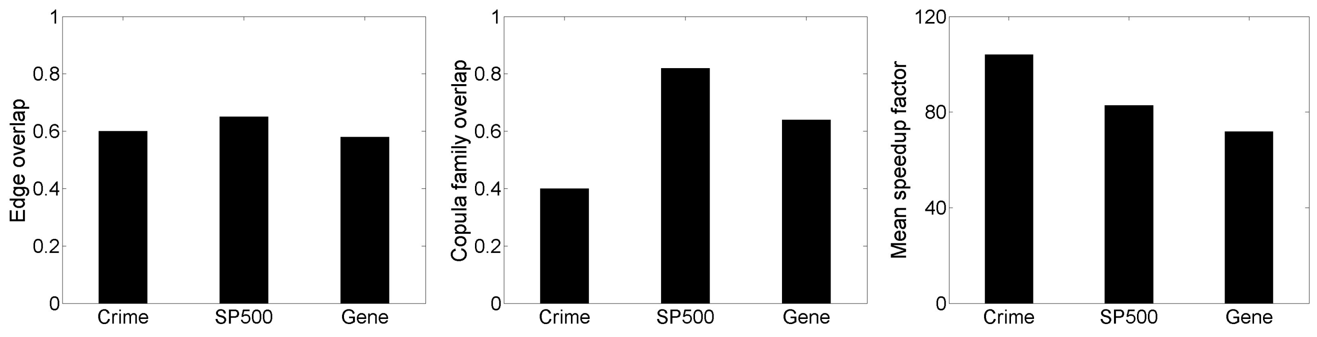


Figure 4: (left) Average fraction of edges that overlap between the model learned using our SMS method and the golden standard model learned using exact maximum likelihood computations. (middle) average fraction of edges, among those common to the two models, that agree on the copula family. (right) average speedup factor of our SMS method over learning with exact computations.

We consider three varied real-life datasets:

- **Crime** (UCI repository). 100 variables relating to crime ranging from household size to fraction of children born outside of a marriage, for 1994 communities across the U.S.
- **SP500**. End of day changes of the value of the 500 stocks (variables) comprising the Standard and Poor's index (S&P 500) over a period of close to 2000 trading days (samples).
- **Gene**. A compendium of gene expression experiments used in [Marion et al., 2004]. We chose genes that have at most one missing experiment. This resulted in 765 variables (genes) and 1088 samples.

Results for all three datasets are reported over 10 random equal splits into train and test samples.

We start by considering the average test log probability per instance, shown in Figure 3(left). The superiority of the mixed family model (white bars) over the Gaussian copula model (gray bar) is clear. Further, in the two bigger datasets, our highly efficient SMS approach improves substantially over the baseline both when taking into account the maximum likelihood golden standard and in absolute terms of bit per instance improvement. Appealingly, as the domain becomes more complex and the number of variable grows, so does our advantage. Figure 3(right) shows a typical more detailed comparison of performance for the **SP500** domain. As can be clearly seen, our superiority is consistently substantial over all random repetitions.

Next, we consider the qualitative ability of our SMS approach that is based on Spearman's ρ_s evaluation to correctly identify both the structure and the best copula family for each edge. Figure 4 (left) shows for each of the datasets the average percentage of edges, over the 10 random runs, that are common to the model learned by our SMS method and the one learned using exact time-consuming computations. As can be clearly seen, the overlap between the trees learned is nontrivial considering the size of the domain the fact that our structure learning approach only relies on simple empirical Spearman's ρ_s estimates. To evaluate the ability of our approach to also correctly choose the right parameterization for each edge, Figure 4 (middle) shows the average percentage of edges that, in addition to being in both the SMS model and that learned using exact computations, also agree on the copula family. Again, given the inherent difficulty of model selection, the similarity between the two models is appealing. The least favorable overlap for the crime domain also explains why our log-probability performance for the crime domain is the least impressive. Still, even in this case, performance is superior to the baseline Gaussian copula only model.

Finally, we consider the speedup factor of our SMS method when learning a mixed family copula model relative to learning using exact computations. Figure 4(right) shows the average speedup factor. For all three domains the speedup is quite impressive at around *two orders of magnitude*. Learning the **SP500** model, for example, takes only minutes on a single CPU making structure learning a significantly more accessible task than in the past. We note that the growth rate of both our SMS method and the exact one as a function of the number of variables is similar, and that the difference is in the dependence on the operations that have to be carried our for each training instance. The speedup reported confirms this since, for example, the **Gene** dataset has almost half the samples of the **Crime** dataset. Thus, while achieving impressive speedups even for the modest datasets considered here, our SMS method is particularly suited to handle a substantial number of training samples.

6 Summary and Future Work

In this paper we addressed the computationally demanding challenge of structure learning for real-valued domains in the context of expressive copula-based models. First, we significantly extended the result of Elidan [2012] and substantiated the conjecture of the monotonic relationship between the magnitude of Spearman's ρ_s and the expected likelihood of an edge in the network. Second, we suggested a novel Bayesian approach for performing structure learning while also allowing for the selection of a different copula family for each edge, without incurring any computational cost. Third, we demonstrated the effectiveness of our SMS approach on varied real-life domains.

Importantly, the domains considered are quite sizable by structure learning standards and dramatically so for copula models. Further, to the best of our knowledge, ours is the first method for automated learning of multivariate copula-based models that allows for a mix of different copula families.

An obvious open theoretical question is the identification of *necessary* conditions for the monotonicity relationship between Spearman's ρ_s and the copula entropy. Another important and practical avenue of research is the exploration of appropriate priors for the density of ρ_s for different copula families. More generally, it would be useful to find other efficient proxies for speedy model selection, e.g., based on other dependence measures such as the Schwizer-Wolff sigma [Schweizer and Wolff, 1981].

7 Appendix

We now prove that the Ali-Mikhail-haq and Gumbel-Barnett copula densities are TP2/RR2, allowing us to apply Theorem 3.3 to these families. We start with a useful property of TP2 that will be used in our proof:

Observation 7.1 : Let $f_1(x,y), f_2(x,y)$ be two real non-negative TP2 (RR2) functions. Then $\Psi(x,y) = f_1 f_2$ is TP2 (RR2).

Proof: From the TP2 property we have

$$\begin{aligned}
&\Psi(x_1,y_1)\Psi(x_2,y_2) = \\
&\quad = \quad f_1(x_1,y_1)f_1(x_2,y_2)f_2(x_1,y_1)f_2(x_2,y_2) \\
&\quad \leq \quad f_1(x_1 \vee x_2, y_1 \vee y_2)f_1(x1 \wedge x_2, y_1 \wedge y_2) \\
&\qquad\quad \times f_2(x_1 \vee x_2, y_1 \vee y_2)f_2(x1 \wedge x_2, y_1 \wedge y_2) \\
&\quad \equiv \quad \Psi(x1 \vee x_2, y_1 \vee y_2)\Psi(x_1 \wedge x_2, y_1 \wedge y_2),
\end{aligned}$$

where the last line follows from the previous by definition after rearranging of terms. The result follows from the definition of a TP2 density. ∎

Lemma 7.2: *The Ali-Mikhail-haq (AMH) density*

$$c(u,v) = \frac{1+\theta[(1+u)(1+v)-3]+\theta^2(\tilde{u})(\tilde{v})}{[1-\theta(\tilde{u})(\tilde{v})]^{-3}},$$

for $\theta \in [-1,1]$ and where $\tilde{u} \equiv (1-u)$, $\tilde{v} \equiv (1-v)$, is TP2 when $\theta \in [0,1]$ and RR2 when $\theta \in [-1,0)$.

Proof: We will show that for $\theta \in [0,1]$, the AMH density is a product of two non-negative TP2 functions, and the result will follow from Observation 7.1:

$$\begin{aligned}
f_1(u,v) &= 1+\theta[(1+u)(1+v)-3]+\theta^2\tilde{u}\tilde{v}, \\
f_2(u,v) &= [1-\theta\tilde{u}\tilde{v}]^{-3}.
\end{aligned}$$

The positivity of f_1, f_2, can be easily verified. Further, a positive function is TP2 iff it is supermodular on the log scale. Also, a function $f(u,v)$ is supermodular iff its second order derivatives are positive, that is $\partial u \partial v f(u,v) \geq 0, (u,v) \in \mathbf{R}^2$. We now have,

$$\frac{\partial^2 \log f_2(u,v)}{\partial u \partial v} = \frac{-3\theta^2\tilde{u}\tilde{v} - \theta[1-\theta\tilde{u}\tilde{v}]}{[1-\theta\tilde{u}\tilde{v}]^2}.$$

This second order derivative is positive so that $f_2(u,v)$ is TP2. Using the same technique, we can show that $\log f_1(u,v)$ is supermoduler, hence $f_1(u,v)$ is also TP2. The proof for $\theta \in [-1,0)$ is similar. ∎

Lemma 7.3: *The Gumbel-Barnett (GB) density by*

$$c(u,v) = (-\theta + [1-\theta\grave{u}][1-\theta\grave{v}])\, e^{-\theta\grave{u}\grave{v}},$$

for $\theta \in (0,1]$, and where $\grave{u} = \theta log(1-u), \grave{v} = \theta log(1-v)$, is an RR2 function.

Proof: We will show that the GB density is a product of two non-negative RR2 functions. Define

$$\begin{aligned}
f_1(u,v) &= -\theta + [1-\theta\grave{u}][1-\theta\grave{v}] \\
f_2(u,v) &= e^{-\theta\grave{u}\grave{v}}
\end{aligned}$$

f_2 is non-negative and since $c(u,v)$ is a density function, f_1 must also be non-negative. Also note that a function is RR2 only if is submodular on the log scale, and that a function is submodular if its second order derivative is non-positive. Starting with $f_1(u,v)$, we have

$$\frac{\partial^2 \log f_1(u,v)}{\partial u \partial v} = \frac{-\theta^3}{(1-u)(1-v)},$$

and this term is always non-positive for $\theta \in (0,1]$. Thus, $f_1(u,v)$ is RR2. The proof that $f_2(u,v)$ is RR2 is similar. It follows that the Gumbel-Barnett density is an RR2 function.

Acknowledgements

This research is funded by ISF Centers of Excellence grant 1789/11, and by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI).

References

Allan.R.Sampson. Positive dependence properties of elliptically symmetric distributions. *Journal of Multivariate Analysis*, 13(2):375–381, 1983.

T. Bedford and R. Cooke. Vines - a new graphical model for dependent random variables. *Annals of Statistics*, 2002.

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.

A.M. Hanea, D. Kurowicka, Roger M. Cooke, and D.A. Ababei. Mining and visualising ordinal data with non-parametric continuous bbns. *Comp Statistics and Data Analysis*, 54(3):668–687, 2010.

H. Joe. Majorization, randomness and dependence for mutivariate distributions. *The Annals of Probability*, 15(3):1217–1225, 1987.

H. Joe. Multivariate models and dependence concepts. *Monographs on Statistics and Applied Probability*, 73, 1997.

S. Kirshner. Learning with tree-averaged densities and distributions. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* The MIT Press, 2009.

D. Kurowicka and R. Cooke. The vine copula method for representing high dimensional dependent distributions: Applications to cont. belief nets. In *Proc. of the Simulation Conf.*, 2002.

R.M. Marion, A. Regev, E. Segal, Y. Barash, D. Koller, N. Friedman, and E.K. O'Shea. Sfp1 is a stress- and nutrient-sensitive regulator of ribosomal protein gene expression. *Proc Natl Acad Sci U S A*, 101(40):14315–22, 2004.

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.

B. Schweizer and E. Wolff. On nonparameteric measures of dependence for random variables. *The Annals of Statistics*, 9, 1981.

M. Shaked and J. Shanthikumar. *Stochastic Orders.* Springer, 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.

Probabilistic inverse reinforcement learning in unknown environments

Aristide C. Y. Tossou
EPAC, Abomey-Calavi, Bénin
yedtoss@gmail.com

Christos Dimitrakakis
EPFL, Lausanne, Switzerland
christos.dimitrakakis@gmail.com

Abstract

We consider the problem of learning by demonstration from agents acting in unknown stochastic Markov environments or games. Our aim is to estimate agent preferences in order to construct improved policies for the same task that the agents are trying to solve. To do so, we extend previous probabilistic approaches for inverse reinforcement learning in known MDPs to the case of *unknown* dynamics or opponents. We do this by deriving two simplified probabilistic models of the demonstrator's policy and utility. For tractability, we use maximum a posteriori estimation rather than full Bayesian inference. Under a flat prior, this results in a convex optimisation problem. We find that the resulting algorithms are highly competitive against a variety of other methods for inverse reinforcement learning that do have knowledge of the dynamics.

1 Introduction

We consider the problem of learning by demonstration from agents acting in stochastic Markov environments, or in stochastic Markov games [13, 18], when we do not know the underlying dynamics of the environment or the opponent strategy. This type of learning is very useful, since it can significantly decrease the time needed to acquire a particular task. Another possible application are games such as chess, where large libraries of expert play have been accumulated over the years. Inverse reinforcement learning offers a principled method to use this data for imitating expert play, even when the experts are deviating from the optimal strategy. In this paper, we learn through demonstration by estimating the preferences and policy of the agent giving the demonstration. These can then be used to estimate improved policies, by combining the inferred agent preferences with policy improvement schemes. This is not always trivial, since the demonstrations may be sub-optimal and the underlying dynamics are unknown to us.

Our main *technical contribution* is two inverse reinforcement learning algorithms for the case of *unknown dynamics*. These are inspired by two Bayesian models for inverse reinforcement learning in known environments [7, 17]. Our first algorithm extends the original model to unknown dynamics by coupling probabilistic preference and policy estimation with approximate dynamic programming schemes and maximum a posteriori estimation. Thus, we avoid explicitly estimating a model altogether. Our second algorithm proposes a simpler model where the policy and value function are jointly represented by the same set of parameters. We can thus eliminate both the dynamic programming step and estimation of the dynamics. Finally, we *experimentally* evaluate our schemes against a broad selection of algorithms which *do* know the dynamics. We consider both agents acting in unknown environments, as well agents playing stochastic games, and show that we can approach and even surpass the performance of methods that use knowledge of the dynamics.

The remainder of the paper is organised as follows. Section 2 formally introduces the setting, while Section 3 discusses related work and our contribution. Section 4 describes our models and algorithms. Comparisons with other methods are given in Section 5, and we conclude with a discussion in Section 6.

2 Setting

In our setting, we observe a set of demonstrations D from an agent acting in an unknown Markov environment or game $\nu \in \mathcal{N}$, where $\mathcal{N}$ is a set of possible environments. The k-th demonstration $d_k \in D$ is a T_k-long trajectory, consisting of a sequence of environment states $s \in \mathcal{S}$ and agent actions $a \in \mathcal{A}$ with

$d_k = \big((s_1^k, a_1^k), \ldots, (s_{T_k}^k, a_{T_k}^k)\big)$. We assume that both the agent policy π and the environment ν are Markovian. Consequently, the action a_t only depends on the current state s_t and the next state s_{t+1} only depends on s_t, a_t. The agent acts according to some (unknown to us) reward function $\rho : \mathcal{S} \times \mathcal{A}$. In particular, any policy π that the agent chooses, has a value defined in terms of the expected utility with respect to ρ:

$$V^{\pi}_{\nu,\rho}(s) \triangleq \mathbb{E}_{\pi,\nu}(U_\rho \mid s_1 = s), \quad U_\rho \triangleq \sum_{t=1}^{T} \rho(s_t, a_t), \tag{2.1}$$

where T may be random.[1] We denote the optimal policy for a reward function by $\pi^*(\nu, \rho)$, with either of those two arguments omitted if it is clear from context. The optimal value function is denoted by:

$$V^*_{\nu,\rho} \triangleq \sup_{\pi} V^{\pi}_{\nu,\rho}. \tag{2.2}$$

Similarly, we define $Q^{\pi}_{\nu,\rho}(s,a) \triangleq \mathbb{E}_{\pi,\nu}(U_\rho \mid s_1 = s, a_1 = a)$ to be the expected utility of taking action a at state s and following π thereafter, while $Q^*_{\nu,\rho}(s,a)$ is the optimal Q-function. We assume that the agent's policy is ε-optimal with respect to ρ, that is:

$$V^{\pi}_{\nu,\rho} \geq V^*_{\nu,\rho} - \varepsilon. \tag{2.3}$$

In our setting, we can observe the sequence of states and actions taken and we know (the distribution of) T. However, the policy π, the environment ν and the reward function ρ, as well as the ε-optimality of the policy are *unknown* to us.

3 Related work and contribution

Inverse reinforcement learning [15] is the problem of estimating the reward function of an agent acting in a dynamic environment. It is thus closely related to preference elicitation [3], since the reward function can be used to calculate the agent preferences, and apprenticeship learning [1], since for each reward function one can calculate the optimal policy.

Much previous work in this setting employs a linear approximation with feature expectations. The general idea is to find a reward function minimising some loss between the expert features expectations and the one of the optimal policy for the approximated dynamics and reward function. A representative example is [8], which uses the quadratic programming maximum-margin approach proposed in [1] to obtain a reward function and least squares temporal differences (LSTD) [4] for policy evaluation. LSTD is also used to estimate the feature expectation of the expert. A similar approach is taken in [14], which used the game-theoretic algorithm MWAL [20] where a near optimal policy is found by using LSPIf, a variant of LSPI [11] for discrete finite horizon Markov decision processes.

Other methods avoid using policy iteration, but nevertheless indirectly employ knowledge of the environment. The most important work in this category is relative entropy inverse reinforcement learning [2]. The main idea is to perform a sub-gradient ascent on an appropriate loss. The sub-gradient is estimated using importance sampling on trajectories generated using a stochastic policy on the real environment. Another work in this category is the SCIRL (Structured Classification for Inverse Reinforcement Learning) algorithm [9]. Using the observation that the reward and the Q function share the same parameter, then estimate it by minimising an appropriately defined loss. While we use a similar idea, SCIRL uses the real environment model to calculate feature expectations.

An altogether different approach is taken by [12], which constructs reward features from a set of component features through logical conjunctions. Given a set of trajectories, and the true environment model, the algorithm then estimates a reward function by finding the most effective feature combinations.

Our algorithms are inspired by two recently proposed probabilistic models for Bayesian inverse reinforcement learning. The first model starts by specifying a prior on the reward function and a reward-conditional distribution on policies through a prior on policy randomness [17]. The second model instead specifies a prior on the policy directly and policy-conditional distribution on the reward functions through a prior on policy optimality [7]. These methods have been described and evaluated on problems where the dynamics are known and the authors suggest that handling the unknown dynamics case would be possible by simply adding a prior distribution on the dynamics.

3.1 Our contribution

Our own work focuses on the problem of learning from demonstration in the case where the transition model of the process is *unknown*. This is the case when the agent is either acting in an environment with unknown dynamics, or playing an alternating Markov game against an unknown opponent. Unlike most previous approaches, we do not have prior knowledge of the dynamics, either in the form of an analytically available transition kernel, or in the form of a simulator from which we can take samples.

With respect to the reward-prior model specified

[1]For geometrically distributed T, the expected utility function is identical to that obtained when $T \to \infty$ and the rewards are exponentially discounted.

in [17], our main technical novelty is to avoid estimating the dynamics altogether. Instead, we use *approximate dynamic programming* (LSTDQ) in combination with linear parametrisation to obtain policies and value functions for the reward prior model.

With respect to the policy-optimality-prior specified in [7], our main technical novelty is that we *eliminate the need* for a dynamic programming step altogether. This significant simplification is achieved by placing a prior on value functions rather than reward functions. Then, instead of considering all possible value functions for which a given policy is ε-optimal, we restrict the space by jointly parametrising the policy and value function.

Thirdly, our models use *features*, rather than the raw state observations. Thus our algorithms are *more generally applicable.*

Finally, for *computational simplicity*, we employ maximum a posteriori (MAP) estimation, rather than full Bayesian inference, as also suggested in [5]. Then inference can be performed quickly using any appropriate global or local optimisation algorithm.

Our main *experimental* contribution is to show that our proposed algorithms are *highly competitive* against six other methods, even if those are using *knowledge* of the dynamics. In particular, we investigate the robustness of our algorithms, by tuning the hyperparameters, including the features, on a fixed set of demonstrations and then testing their performance on a large sample of demonstrations drawn from some distribution.

We compare our approaches with the full Bayesian method RPB in [17], the MAP$_{\text{IRL}}$ method in [5], MWAL [20], the maximum entropy method MaxEnt [21], the projection method Proj suggested in [1], MMP [16] and finally the feature-based FIRL [12]. We outperform most of those methods in all of our experiments, apart from the FIRL which (naturally) had particularly good performance in a domain with a high number of features. However, even in this case, our algorithms, without any knowledge of the dynamics, manage to approach the FIRL solution.

4 Models

The algorithms we provide are inspired by the probabilistic models introduced in [7, 17], whose difference from our models we describe in this section. Both of these maintain a joint prior distribution ξ on policies $\pi \in \mathcal{P}$ and reward functions $\rho \in \mathcal{R}$. They only differ on how they model their dependencies. The *reward prior* model specifies a prior on reward functions and a conditional distribution on policies given the reward. The *policy optimality* model specifies a prior on policies and a conditional distribution on reward functions given the policy. In both cases, the likelihood is simply the probability $p(D \mid \pi)$ of observing the demonstrations under the estimated policy.

Both of our models use features, rather than direct state observations. Thus, we define a mapping $g_R : \mathcal{S} \times \mathcal{A} \to \mathcal{X}_R$ from the state-action space to a feature space $\mathcal{X}_R \subset \mathbb{R}^{m_R}$ for the reward function. Similarly, we also define a mapping $g_Q : \mathcal{S} \times \mathcal{A} \to \mathcal{X}_Q$ from the state-action space to a feature space $\mathcal{X}_Q \subset \mathbb{R}^{m_Q}$ for the state-action value function.

4.1 Reward prior (RP) model

The main difference between the original model and ours is with respect to the reward function parametrisation and the value function estimation. In our model, we maintain a parameter $\boldsymbol{w}_R \in \mathbb{R}^{m_R}$ parametrising the reward function via a linear function $f : \mathcal{X}_R \times \mathbb{R}^{m_R} \to \mathbb{R}$, such that:

$$\rho(s, a) \triangleq f(g_R(s, a), \boldsymbol{w}_R) \triangleq g_R(s, a)^\top \boldsymbol{w}_R. \tag{4.1}$$

Since the environment dynamics are not known, we employ least-square temporal differences (LSTDQ [11]) to obtain a parametrised state-action value function:

$$\hat{Q}^*(s, a) = g_Q(s, a)^\top \boldsymbol{w}_Q. \tag{4.2}$$

The use of LSTDQ allows us to consider the case where the demonstration D comes from a suboptimal expert. Since D is generated by the expert's policy, we can use the on-policy LSTDQ. This finds a parameter $\boldsymbol{w}_Q$ such that $\boldsymbol{w}_Q = A^{-1}b = A^{-1}Z\boldsymbol{w}_R = C\boldsymbol{w}_R$, with $C = A^{-1}Z$ and :

$$A = \sum_{d_k \in \mathcal{D}} \sum_{t=1}^{T_k - 1} g_Q(s_t^k, a_t^k) \left(g_Q(s_t^k, a_t^k) - \gamma g_Q(s_{t+1}^k, a_{t+1}^k) \right)^\top \tag{4.3}$$

$$\begin{aligned} b &= \sum_{d_k \in \mathcal{D}} \sum_{t=1}^{T_k - 1} g_Q(s_t^k, a_t^k) \rho(s_t^k, a_t^k) && (4.4) \\ &= \sum_{d_k \in \mathcal{D}} \sum_{t=1}^{T_k - 1} g_Q(s_t^k, a_t^k) g_R(s_t^k, a_t^k)^\top \boldsymbol{w}_R && (4.5) \\ &= \left(\sum_{d_k \in \mathcal{D}} \sum_{t=1}^{T_k - 1} g_Q(s_t^k, a_t^k) g_R(s_t^k, a_t^k)^\top \right) \boldsymbol{w}_R && (4.6) \\ &= Z\boldsymbol{w}_R. && (4.7) \end{aligned}$$

So:

$$Z = \sum_{d_k \in \mathcal{D}} \sum_{t=1}^{T_k - 1} g_Q(s_t^k, a_t^k) g_R(s_t^k, a_t^k)^\top. \tag{4.8}$$

As the demonstration D is fixed, so are A, Z and C. Then,

$$\hat{Q}^*_{\boldsymbol{w}_R}(s,a) = g_Q(s,a)^\top C\boldsymbol{w}_R. \tag{4.9}$$

As in the original model, we assume that, the demonstration policy π is *softmax* with respect to the value function:

$$\pi_{\beta,\boldsymbol{w}_R}(a \mid s) \triangleq \frac{e^{\beta \hat{Q}^*_{\boldsymbol{w}_R}(s,a)}}{\sum_{a'} e^{\beta \hat{Q}^*_{\boldsymbol{w}_R}(s,a')}}, \tag{4.10}$$

Finally, the likelihood is simply:

$$p(D \mid \beta, \boldsymbol{w}_R) = \prod_{d_k \in \mathcal{D}} \prod_{t=1}^{T_k} \pi_{\beta,\boldsymbol{w}_R}(a_t^k \mid s_t^k), \tag{4.11}$$

and the overall posterior can be factorised due to conditional independence:

$$\xi(\beta, \boldsymbol{w}_R \mid D) \propto p(D \mid \beta, \boldsymbol{w}_R)\xi(\boldsymbol{w}_R)\xi(\beta). \tag{4.12}$$

The main question is which prior to select. If we assume a uniform prior for the reward parameters $\boldsymbol{w}_R$ and β, taking the logarithm and replacing $\hat{Q}^*_{\boldsymbol{w}_R}$ by its value results in the following maximisation problem:

$$\max_{\boldsymbol{w}_R} \sum_{d_k \in \mathcal{D}} \Big\{ \sum_{t=1}^{T_k} \beta g_Q(s_t^k, a_t^k)^\top C\boldsymbol{w}_R - \ln \sum_{a'} e^{\beta g_Q(s_t^k,a')^\top C\boldsymbol{w}_R} \Big\}. \tag{4.13}$$

Equation 4.13 is a concave function which can be maximised efficiently. Note that since $\boldsymbol{w}_R$ is not constrained and the prior for $\boldsymbol{w}_R, \beta$ is uniform, β is essentially a free parameter which can be set to an arbitrary positive value. Then, we can get both the preference of the expert $\boldsymbol{w}_R$, as well as the parameter of the value function $\boldsymbol{w}_Q$.

An *alternative* is to use an exponential prior for β and a Dirichlet prior for ρ. Intuitively, this should induce a penalty for overfitting, and a sparse reward function. However, the optimisation is in this case difficult, and in preliminary experiments we found that it did not improve upon the uniform prior.

4.2 Policy optimality (PO) model

This model starts with a prior $\xi(\pi)$ on the policies. This allows a posterior on the policy $\xi(\pi \mid D)$ to be obtained from the data directly. This type of "imitator" policy is then tempered through use of a prior on *policy optimality.* Briefly, if the policy is ε-optimal, then there is a set of reward functions $\mathcal{R}_\varepsilon$ such that $V^\pi_{\nu,\rho} \geq V^*_{\nu,\rho} - \varepsilon$ for any $\rho \in \mathcal{R}_\varepsilon$.

The *original* model assumed a prior $\xi(\varepsilon) = e^{-\varepsilon}$, for ε-optimality, and so obtained a distribution on reward functions conditional on the policy and its optimality. The overall posterior was then factorised as follows:

$$\xi(\varepsilon, \pi, \rho \mid D, \nu) \propto p(D \mid \pi, \nu)\xi(\rho \mid \pi, \varepsilon)\xi(\varepsilon)\xi(\pi). \tag{4.14}$$

The difficulty with this model is that one has to consider all reward functions for which a policy is ε-optimal. This is not a problem for a finite set reward functions, but it makes inference hard in the general case. In addition, since ν is not known, one would have to integrate (or maximise) over ν explicitly.

Our own model considers *value functions* directly, rather than reward functions, thus eliminating the need for a dynamic programming step. In particular, a vector $(\boldsymbol{w_Q}, \beta)$ with $\boldsymbol{w_Q} \in \mathbb{R}^{m_Q}$, jointly parametrises the optimal value function and the demonstrator policy. Specifically, the optimal state-action value function is parametrised via a linear function $h : \mathcal{X}_Q \times \mathbb{R}^{m_Q} \to \mathbb{R}$, such that:

$$Q(s,a) = h(g_Q(s,a), \boldsymbol{w_Q}), \tag{4.15}$$

while the policy is defined as:

$$\pi_{\beta,\boldsymbol{w_Q}}(a \mid s) = \frac{e^{\beta Q(s,a)}}{\sum_{a' \in \mathcal{A}} e^{\beta Q(s,a')}}. \tag{4.16}$$

Each parameter $\beta, \boldsymbol{w_Q}$ corresponds to a unique policy-value function pair. We assume an independent prior $\xi(\beta, \boldsymbol{w_Q}) = \xi(\beta)\xi(\boldsymbol{w_Q})$. Our posterior probability is then:

$$\xi(\beta, \boldsymbol{w_Q} \mid D) \propto p(D \mid \pi_{\beta,\boldsymbol{w_Q}})\xi(\beta)\xi(\boldsymbol{w_Q}), \tag{4.17}$$

since given $\beta, \boldsymbol{w_Q}$, the policy and value function are uniquely determined.

Using uniform priors for all parameters, and taking the logarithm, results in the following maximisation problem:

$$\max_{\boldsymbol{w}_Q} \sum_{d_k \in \mathcal{D}} \left\{ \sum_{t=1}^{T_k} \beta g_Q(s_t^k, a_t^k)^\top \boldsymbol{w}_Q - \ln \sum_{a'} e^{\beta g_Q(s_t^k,a')^\top \boldsymbol{w}_Q} \right\}. \tag{4.18}$$

It is easy to see that PO model can also be obtained by replacing $C\boldsymbol{w}_R$ in equation 4.13 by the value $\boldsymbol{w}_Q$. Although both PO and RP look similar, the PO model does not require the matrix C. Intuitively we can see that PO considers value functions directly, rather than reward functions, thus eliminating the need for a dynamic programming step. In particular, a vector $\boldsymbol{w}_Q$ jointly parametrises the optimal value function and the demonstrator policy defined in equation 4.10. Another significant difference is that PO starts with a prior $\xi(\pi)$ on the policies. This allows a posterior on the policy $\xi(\pi \mid D)$ to be obtained from the data directly.

An *alternative model* is to use an exponential distribution, $\xi(\beta^{-1}) = \mathcal{Exp}(1)$ for the scaling parameter, and a Dirichlet distribution $\xi(\boldsymbol{w_Q}) = \mathcal{Dirichlet}(\boldsymbol{\alpha})$ for the other value function parameters, with $\boldsymbol{\alpha} \in \mathbb{R}_+^{m_Q}$ such that $\alpha_i = \frac{1}{2}$, so the posterior is:

$$\xi(\beta, \boldsymbol{w_Q} \mid D) \propto \prod_t \pi_{\beta,\boldsymbol{w_Q}}(a_t \mid s_t) \prod_{i=1}^{m_Q} w_{Qi}^{\alpha_i - 1} e^{-\beta^{-1}}. \tag{4.19}$$

Taking the logarithm, we now need to solve the following maximisation problem, under the constraint $\|\boldsymbol{w_Q}\|_1 = 1$, and $w_{Qi} \in [0, 1]$:

$$\max_{\beta, \boldsymbol{w_Q}} \sum_t \ln \pi_{\beta,\boldsymbol{w_Q}}(a_t \mid s_t) - \frac{1}{2} \sum_{i=1}^{m_Q} \ln w_{Qi} - \beta^{-1}. \tag{4.20}$$

Now β^{-1} can be seen as a simple penalty to avoid overfitting. Due to the constraints, parameters $\boldsymbol{w_Q}$ where a few components have large values are favoured. Thus, this model can be seen as a sparse regularised classifier.

5 Experiments

We performed a set of experiments in three domains. These are either stochastic environments or alternating Markov games. In all those experiments, our algorithms did not assume knowledge of the underlying process, or opponent. For those cases where it was possible to obtain an *analytic* model of the process and/or of the opponent, we also compared our algorithms with methods which assume knowledge of the underlying dynamics. In all cases, we first tuned hyper-parameters of the algorithms in a small set of demonstration trajectories. These algorithms were then evaluated on multiple runs with varying amounts of demonstration trajectories, in order to examine the robustness of algorithms and their performance improvement as the amount of data increased.

As mentioned in the previous section, we have a choice of different priors for the models. The uniform priors lead to convex problems, which allow us to use an efficient algorithm (L-BFGS) to find the maximum a posteriori parameters. In preliminary experiments with the alternative priors, we had to resort to random restarts combined with local search. This didn't led to better results, possibly due to the difficulty of the optimisation problem. Consequently, in the presented experiments, we only show results with uniform priors. We use LPO to denote the linear PO model and LRP for the linear RP model.

All the experiments are performed on discrete environments, even though inference is performed on a set of features instead. Consequently, it is possible to compute the optimal policy with respect to a given reward function on all of these environments. In our experiments, we compare the performance of the policies found by the inverse RL algorithms to that of the optimal policy of the environment. In particular, the loss of a policy π is:

$$\ell(\pi) \triangleq \sum_{s \in \mathcal{S}} \mu(s)(V^*(s) - V^\pi(s)), \tag{5.1}$$

where μ is the starting state distribution of the problem.

In all these experiments, the features are appropriately scaled for each algorithm. For example, when using MWAL the features are scaled so that they lie in $[-1, 1]$, whereas for Proj, they lie in $[0, 1]$. For all experiments the features of the reward function g_R are state features.

5.1 Blackjack

The first domain is blackjack, as described in [19]. The goal is to get cards such that their sum is as close as possible to 21 without exceeding it. All face cards count as 10 and the ace can be either 1 or 11 (*usable*). At the beginning of the game, two cards are dealt to both the dealer and the player. One of the card of the dealer is face up, the other is face down. If the player has immediately 21 (a face and an ace), then it is a *natural* and he wins (+1.5 points). Otherwise, he can request additional cards one by one (*hits*) until he either stops (*sticks*) or exceeds 21 (*goes bust*). If he goes bust, he loses (-1 points); if he sticks, then it is the dealer's turn. The dealer sticks when he has 17 or greater; he hits otherwise. If the dealer goes bust, then the player wins (+1 points); otherwise the outcome is determined by who ever is closer to 21.

The state is determined by the sum of the player's hand (12-21), the dealer card (ace-10 or 1-10), and whether or not the player holds a usable ace. We ignore the case where the player sum is less than 12, as then the player will always hit. This results in 201 states, including the terminal state.

We used 14 features for the reward function g_R. They included the state variables, their higher order (multivariate) polynomial terms (up to 2) and a bias (which is always 1). For the value function features g_Q, we repeated the reward features for each action. In this domain, the initial state distribution μ is derived from the uniform distribution on the set of cards.

The demonstrations are obtained from the optimal policy. We varied the number of episodes (from 10 to 10000) to determine the ability of the different algorithms to handle limited data. We report average performance on 200 runs. This includes both the loss (5.1) and CPU time.

5.2 Gridworld

The second experiment is on 32×32 gridworlds. The agent can move into 5 directions (west, east, north, south, or be still). But with probability 0.3 it fails and moves to a random direction. The grids are partitioned into 3 non overlapping regions (one at the lower left corner, one at the upper right corner and the third elsewhere). The initial state is uniformly drawn from the possible states. The true reward is positive in the two regions at the corner and negative elsewhere.

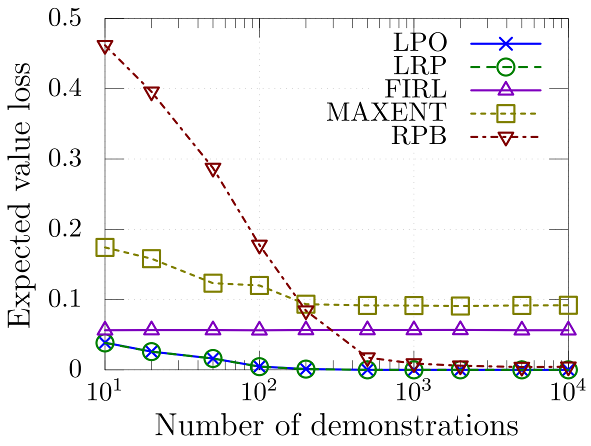
We used 64 features for the reward function g_R. These features included the coordinates x and y and 62 binary features indicating if x or y is lower than some value. For the value function features g_Q, we repeated the reward features for each action.

The demonstrations are obtained from an optimal agent. The number of steps for each episode is 8. The number of episodes is varied from 10 to 10000. We measured the average performance loss with respect to the optimal value function, over 10 experimental runs.

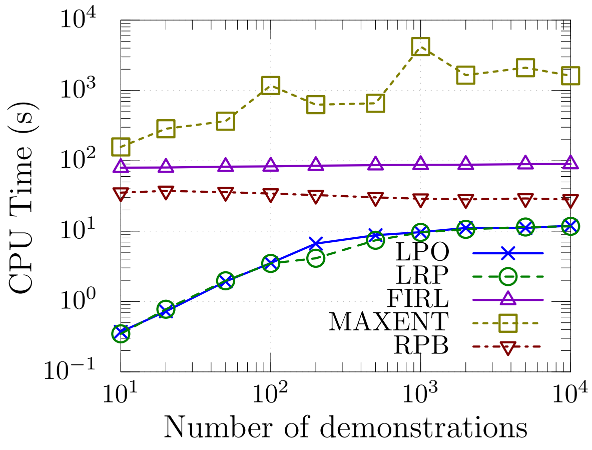
5.3 Tic-Tac-Toe

The final experiment is the game of tic-tac-toe. It is a 3×3 board game where two players (X and O) take turns alternatively to place their mark on empty spaces. The player who first places three marks in a row, horizontally, vertically, or diagonally wins and obtains a reward of $+1$. If there is no legal move remaining, the game is a draw. The game state can be described by 9 factors, each indicating the mark at a board location. Thus the total number of states is at most 3^9, symmetry and unreachable states notwithstanding. There up to 9 possible actions.

To construct g_R, we used the features described in [10]: the number of singlets (horizontal, diagonal, vertical lines with exactly one symbol X or O), doublets (horizontal, diagonal, vertical lines with exactly two symbols X or O), triplets (horizontal, diagonal, vertical lines with exactly three symbols X or O), diversity (the number of different singlet directions for each player) and crosspoints (an empty field belonging to at least two singlets of the same player) for each player; their



(a) Loss (5.1)



(b) CPU time

Figure 1: Blackjack results.

higher order (multivariate) polynomial terms (up to 2) as well as the 9 features for the raw board position.

The features g_Q of the value function for a state-action pair (s, a) are those of the resulting state s' prior to the opponents play. We learned the policy for player X. The demonstrations are obtained from a game between the optimal player (X) and a uniformly random player (O). Player X randomly selects one of the available minimax-optimal actions. We performed the experiments by varying the number of demonstrations (from 10 to 10000) and report results averaged over 200 runs. We evaluated two cases: one against a random player and one against an optimal player.

5.4 Results

For clarity of presentation, the figures only compare our algorithms with the original reward prior model RPB, the maximum entropy method MAX-ENT [21] and the feature-based FIRL [12]. While results for

MWAL [20], MAP$_{\text{IRL}}$ [5], MMP [16] and PROJ [1], are omitted for clarity of presentation, they did not in general perform better than the methods shown. Unlike our algorithms, all of the methods we compare against use knowledge of the environment and consequently enjoy an additional advantage.

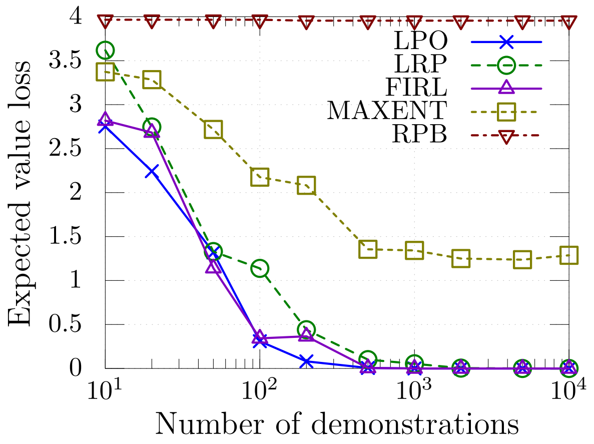
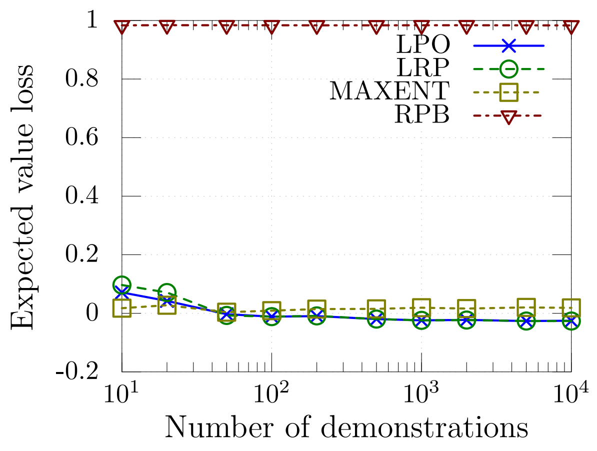


Figure 2: Gridworld results.

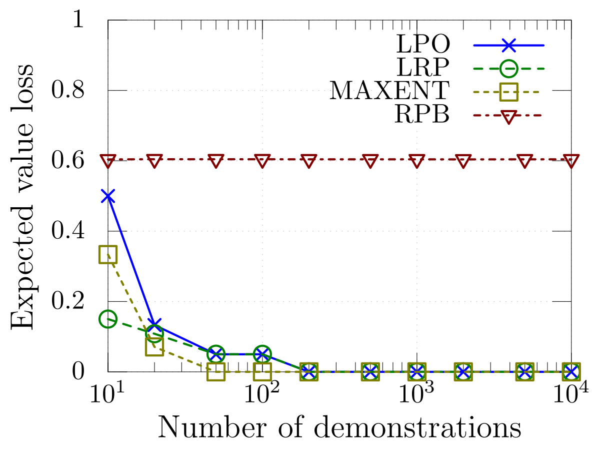
Figure 1(a) shows the expected loss ℓ for **blackjack** for LPO, LRP and prior algorithms. We can see that most algorithms found a good policy as the amount of data increases. However, RPB, MMP and MWAL (not shown) completely failed to find a reward inducing a policy close to the expert's one (always hits) even after 10^4 demonstrations. The performance of MAX-ENT is better but does not improve with more data, in contrast to the original Bayesian approach[2] RPB, and the maximum a posteriori approach MAP$_{\text{IRL}}$. The best competing approach is FIRL, but LRP manages to surpass its performance, even without knowledge of the dynamics. In addition, both LPO and LRP require significantly less computation time than all other approaches, as can be seen in Fig. 1(b).

Figure 2 presents the expected value loss on the **gridworld**. The performance of LRP, LPO, and FIRL increases with more demonstrations. With less data, we can see that LPO slightly outperforms all other algorithms including FIRL. Again, the performance of MAX-ENT does not improve with more data. Also the performance of the original Bayesian approach RPB does not anymore increases with more data.

As a final example, we also present results on **tic-tac-toe** in Fig 3. This way, one may obtain an idea of how well our methods might be applicable to larger games. Figure 3(a) presents the value difference when



(a) Random opponent



(b) Optimal opponent

Figure 3: Tic-Tac-Toe results

the opponent is random opponent, while Figure 3(b) shows the results when the opponent is non deterministic but optimal. Against an optimal opponent, LPO, LRP and MAX-ENT quickly converge to the optimal (minimax) policy. Against a random opponent both LPO and LRP were able to outperform the optimal (minimax) expert while MAX-ENT did not improve from the expert. This can be explained by the fact that MAX-ENT assumes that the expert is optimal and then try to find a reward function which makes the expert optimal. In constrast, LPO and LRP considers a suboptimal expert (ϵ-optimal expert), then try to find the true reward of this expert in order to outperform it if possible. Intuitively, the policy found by LPO and LRP takes a little risk in order to beat the expert against random opponents. At the same time, the found policy never looses against an optimal (minimax) expert. In this experiment, FIRL (not shown) completely failed to find a good policy. Its curve is y=1 for both the random or the optimal opponent.

[2]We also compared our method against the other models detailed in [7, 17], but RPB gave the best results in the tested domains.

Overall, our algorithms are extremely robust and perform near-optimally in all domains. In the blackjack domain they reach the performance of the optimal policy and surpass all alternatives, all of which know the environment model *a priori*. In the gridworld, they compete well against FIRL and even outperformed it with less data in spite of the sophisticated feature discovery method and the knowledge of the dynamics by FIRL. Finally, the tic-tac-toe illustration demonstrates that such methods may also be useful in larger games.

Finally, it should be noted that our algorithms always outperform methods using a linear combination of the features. This could be explained by the fact that our cost function is convex and thus we are guaranteed to find the global solution. This is supported by our results for the fully Bayesian RPB method, as well as our preliminary results using different priors, which result in a non-convex problem.

6 Conclusion

We have provided a set of MAP approaches for probabilistic inverse reinforcement learning in unknown environments. These approaches are inspired by probabilistic models originally presented in [7, 17]. By avoiding full Bayesian inference, we were able to extend these methods to the case of unknown dynamics, Markov games and feature observations instead of actual state-action observations. In addition, we proposed an interesting simplification to the policy optimality model presented in [7], such that the set of all ϵ-optimal reward functions for a particular policy does not need to be searched for the policy optimality algorithm.

Experimentally, both algorithms have a performance which equals and even surpasses that of algorithms which assume some knowledge of the underlying dynamics. This includes both approaches which use the dynamics analytically and approaches which only sample from the dynamics. In addition, the computational requirements of both methods are modest and significantly lower than those of the alternatives we tried.

One question is whether it is possible or useful to extend these algorithms to the nonlinear case. This can be done by using some kind of kernel, thus preserving the linearity in the features. This could be of interest for applications where the reward is complex with respect to the features. Finally, we would like to apply these methods to larger problems, and in particular to learning to play games from databases of expert play. It would be interesting to do so while extending them to multiple reward functions, such as for the models considered in [6, 7].

Acknowledgements

We wish to thank the anonymous reviewers for their comments on this as well as an earlier version of this paper. This work was partially supported by the Marie Curie Project ESDEMUU, Grant Number 237816.

References

[1] P. Abbeel and A.Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st international conference on Machine learning (ICML 2004)*, 2004.

[2] Abdeslam Boularias, Jens Kober, and Jan Peters. Relative entropy inverse reinforcement learning. In *AISTATS 2011, Journal of Machine Learning Research - Proceedings Track*, volume 15, pages 182–189, 2011.

[3] C. Boutilier. A POMDP formulation of preference elicitation problems. In *Proceedings of the National Conference on Artificial Intelligence*, pages 239–246. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2002.

[4] S.J. Bradtke and A.G. Barto. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22(1):33–57, 1996.

[5] J. Choi and K.E. Kim. MAP inference for Bayesian inverse reinforcement learning. In *NIPS*, pages 1989–1997, 2011.

[6] J. Choi and K.E. Kim. Nonparametric Bayesian inverse reinforcement learning for multiple reward functions. In *NIPS*, 2012.

[7] Christos Dimitrakakis and Constantin A. Rothkopf. Bayesian multitask inverse reinforcement learning. In *European Workshop on Reinforcement Learning (EWRL 2011)*, number 7188 in LNCS, pages 273–284, 2011.

[8] Edouard Klein, Matthieu Geist, and Olivier Pietquin. Batch, Off-policy and Model-free Apprenticeship Learning. In *Proceedings of the 9th European Workshop on Reinforcement Learning*, pages 1–12, Athens, Greece, September 2011.

[9] Edouard Klein, Matthieu Geist, Bilal Piot, and Olivier Pietquin. Inverse Reinforcement Learning through Structured Classification. In *Advances in Neural Information Processing Systems (NIPS 2012)*, Lake Tahoe (NV, USA), December 2012.

[10] Wolfgang Konen and Thomas Bartz-Beielstein. Reinforcement learning for games: failures and successes. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, pages 2641–2648. ACM, 2009.

[11] M.G. Lagoudakis and R. Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

[12] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Feature construction for inverse reinforcement learning. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1342–1350. 2010.

[13] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163. Morgan Kaufmann, 1994.

[14] Takeshi Mori, Matthew Howard, and Sethu Vijayakumar. Model-free apprenticeship learning for transfer of human impedance behaviour. In *IEEE Int. Conf. Humanoid Robots*, 2011.

[15] Andrew Y. Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conf. on Machine Learning*, pages 663–670. Morgan Kaufmann, 2000.

[16] Nathan D. Ratliff, Andrew J. Bagnell, and Martin A. Zinkevich. Maximum margin planning. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 729–736, New York, NY, USA, 2006.

[17] Constantin A. Rothkopf and Christos Dimitrakakis. Preference elicitation and inverse reinforcement learning. In *ECML/PKDD (3)*, volume 6913 of *LNCS*, pages 34–48, 2011.

[18] L. S. Shapley. Stochastic games. *PNAS*, pages 1095–1100, 1953.

[19] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* 1998.

[20] Umar Syed and Robert E. Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in Neural Information Processing Systems*, volume 10, 2008.

[21] Brian D. Ziebart, J. Andrew Bagnell, and Anind K. Dey. Modelling interaction via the principle of maximum causal entropy. In *Proceedings of the 27th International Conference on Machine Learning (ICML 2010)*, Haifa, Israel.

Approximate Kalman Filter Q-Learning for Continuous State-Space MDPs

Charles Tripp
Department of Electrical Engineering
Stanford University
Stanford, California, USA
cet@stanford.edu

Ross Shachter
Management Science and Engineering Dept
Stanford University
Stanford, California, USA
shachter@stanford.edu

Abstract

We seek to learn an effective policy for a Markov Decision Process (MDP) with continuous states via Q-Learning. Given a set of basis functions over state action pairs we search for a corresponding set of linear weights that minimizes the mean Bellman residual. Our algorithm uses a Kalman filter model to estimate those weights and we have developed a simpler approximate Kalman filter model that outperforms the current state of the art projected TD-Learning methods on several standard benchmark problems.

1 INTRODUCTION

We seek an effective policy via Q-Learning for a Markov Decision Process (MDP) with a continuous state space. Given a set of basis functions that project the MDP state action pairs onto a basis space, a linear function maps them into Q-values using a vector of basis function weights. The challenge in implementing such a projection-based value model is twofold: to find a set of basis functions that allow a good fit to the problem when using a linear model, and given a set of basis functions, to efficiently estimate a set of basis function weights that yields a high-performance policy. The techniques that we present here address the second part of this challenge. For more background on MDPs, see (Bellman, 1957, Bertsekas, 1982, Howard, 1960) and for Q-Learning see (Watkins, 1989).

In **Kalman Filter Q-Learning** (KFQL), we use a Kalman filter (Kalman, 1960) to model the weights on the basis functions. The entire distribution over the value for any state action pair is captured in this model, where more credible assessments will yield distributions with smaller variances. Because we have probability distributions over the weights on the basis functions, and over the observations conditioned on those weights, we can compute Bayesian updates to the weight distributions. Unlike the current state of the art method, Projected TD Learning (PTD), Kalman Filter Q-Learning adjusts the weights on basis functions more when they are less well known and less when they are more well known. The more observations are made for a particular basis function, the more confident the model becomes in its assessment of its weight.

We simplify KFQL to obtain **Approximate Kalman Filter Q-Learning** (AKFQL) by ignoring dependence among basis functions. As a result, each iteration is linear in the number of basis functions rather than quadratic, but it also appears to be significantly more efficient and robust for policy generation than either PTD or KFQL.

In the next section we present KFQL and AKFQL, followed by sections on experimental results, related work, and some conclusions.

2 KALMAN FILTER Q-LEARNING

In each time period an MDP has a continuous state s and an action a is chosen from a corresponding set of actions A_s. The transition probabilities to a successor state s' are determined by the state action pair (s, a), and the reward $R(s, a, s')$ for that period is determined by the two states and the action. We seek to learn an optimal policy for choosing actions and the corresponding optimal net present value of the future rewards given that we start with state action pair (s, a), the Q-value $Q(s, a)$. To make that analysis tractable it is standard to introduce basis functions $\phi(s, a)$.

Given n basis functions $\phi(s, a)$ for MDP state action pairs (s, a), we learn an n-vector of weights r for those basis functions that minimizes the mean Bellman residual. Our prior belief is that r is multivariate normal with n-vector mean μ and $n \times n$ covariance matrix Σ. We model the Q-value by $Q(s, a) = r^T \phi(s, a)$

with predicted value $\mu^T\phi(s,a)$ and variance $\sigma^2(s,a)$. We then treat the Q-Learning sample update $\nu(s,a)$ as an observation with variance $\epsilon(s,a)$, which we assume is conditionally independent of the prediction given r. Under these assumptions we can update our beliefs about r using a Kalman filter. This notation is summarized in Table 1.

Table 1: Symbol Definitions

Symbol	**Definition**
s	current MDP state
s'	successor MDP state
$a \in A_s$	action available in state s
$R(s,a,s')$	MDP reward
γ	MDP discount rate
$\phi(s,a)$	basis function values for state action pair (s,a)
r	weights on the basis functions that minimize the mean Bellman residual
μ, Σ	mean vector and covariance matrix for $r \sim N(\mu,\Sigma)$
$Q(s,a)$	Q-value for (s,a)
$\sigma^2(s,a)$	variance of $Q(s,a)$
$\nu(s,a)$	Q-Learning sample update value for (s,a)
$\epsilon(s,a)$	variance of observation $\nu(s,a)$, also called the **sensor noise**.

Kalman Filter Q-Learning, as discussed here, is implemented using the covariance form of the Kalman filter. In out experiments the covariance form was less prone to numerical stability issues than the precision matrix form. We also believe that dealing with covariances makes the model easier to understand and to assess priors, sensor noise, and other model parameters. While the update equations for the KFQL are standard, the notation is unique to this application. We describe the exact calculations involved in updating the KFQL below.

2.1 THE KALMAN OBSERVATION UPDATE

The generic Kalman filter **observation update** equations can be computed for any $r \sim N(\mu,\Sigma)$ with predicted measurement with mean $\mu^T\phi$ and variance $\phi^T\Sigma\phi$. The observed measurement ν has variance ϵ. First we compute the **Kalman gain** n-vector,

$$G = \Sigma\phi(\phi^T\Sigma\phi + \epsilon)^{-1}. \quad (1)$$

The Kalman gain determines the relative impact the update makes on the elements of μ and Σ.

$$\mu \leftarrow \mu + G(\nu - \mu^T\phi) \quad (2)$$
$$\Sigma \leftarrow (I - G\phi^T)\Sigma \quad (3)$$

The posterior values for μ and Σ can then be used as prior values in the next update. For in-depth descriptions of the Kalman Filter see these excellent references: (Koller and Friedman, 2009, Russell and Norvig, 2003, Welch and Bishop, 2001).

2.1.1 MDP Updates

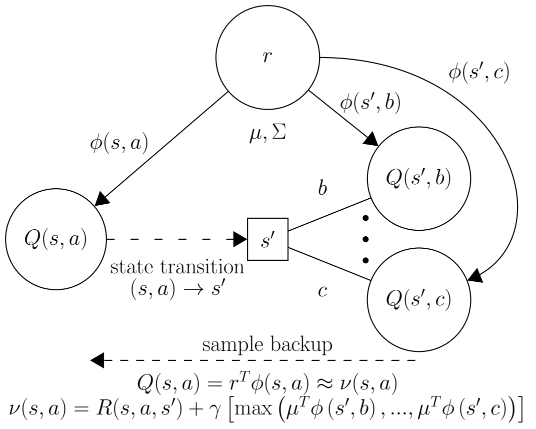

r
$\phi(s',c)$
$\phi(s',b)$
$\phi(s,a)$
μ,Σ
$Q(s',b)$
b
$Q(s,a)$
s'
state transition
$(s,a) \rightarrow s'$
c
$Q(s',c)$
sample backup
$Q(s,a) = r^T\phi(s,a) \approx \nu(s,a)$
$\nu(s,a) = R(s,a,s') + \gamma\left[\max\left(\mu^T\phi\left(s',b\right),...,\mu^T\phi\left(s',c\right)\right)\right]$

Figure 1: The Sample Update

To update our distribution for r, we estimate the value of a sampled successor state s' and use that value in our observation update. Because this method is similar to the Bellman residual minimizing approximation used in Q-Learning, we will refer to it as the **sample update**. It is shown in Figure 1. To arrive at the update formulation, we start with the Bellman equation:

$$Q(s,a) \approx R(s,a,s') + \gamma \max_{a' \in A_{s'}} Q\left(s',a'\right) \quad (4)$$

and then we apply the Q-value approximation $Q(s,a) = r^T\phi(s,a)$, to get:

$$\begin{aligned} Q(s,a) &= r^T\phi(s,a) && (5)\\ &\approx R(s,a,s') + \gamma E\big[\max_{a' \in A_{s'}} r^T\phi(s',a')\big] && (6)\\ &\approx R(s,a,s') + \gamma \max_{a' \in A_{s'}} \mu^T\phi(s',a') && (7)\\ &= \nu(s,a) && (8)\end{aligned}$$

There are several approximations implicit in this update in addition to the Bellman equation. First, we assume that the Q-value $Q(s,a)$ can be represented by $r^T\phi(s,a)$. Second, we interchange the expectation and maximization operations to simplify the computation.

Finally, we do not account for any correlation between the measurement prediction and the sample update, both of which are noisy functions of the weights on the basis functions r. Instead, we estimate the prediction error $\sigma^2(s,a) = \phi(s,a)^T \Sigma \phi(s,a)$ separately from the sensor noise $\epsilon(s,a)$ as explained below.

2.1.2 Computing the Sensor Noise

The sensor noise, $\epsilon(s,a)$, plays a role similar to that of the learning rate in TD Learning. The larger the sensor noise, the smaller the effect of each update on μ. The sensor noise has three sources of uncertainty. First, the sensor suffers from inaccuracies inherent in the model because it is unlikely that the Q-values for all state action pairs can be fit exactly for any value of r. Because this source of noise is due to the model used, we will refer to it as **model bias**. Second, there is sensor noise due to the random nature of the state transition because the value of the sample update depends on which new state s' is sampled. We will refer to this source of noise as **sampling noise**. Both model bias and sampling noise are typically modeled as constant throughout the process, and ϵ_0 should be chosen to capture that combination. Finally, there is sensor noise from the assessments used to compute ν because our assessment of $Q(s',a')$ has variance $\sigma^2(s',a') = \phi(s',a')^T \Sigma \phi(s',a')$. We will refer to this variance as **assessment noise**. Assessment noise will generally decrease as the algorithm makes more observations. All together, the sample update is treated as a noisy observation of $Q(s,a)$ with sensor noise $\epsilon(s,a)$.

We have tried several heuristic methods for setting the sensor noise and found them effective. These include using the variance of the highest valued alternative (Equation 9), using the average of the variances of each alternative (Equation 10), using the largest variance (Equation 11), and using a Boltzmann-weighted average of the variances (Equation 12). Although each of these techniques appears to have similar performance, as is shown in the next section, the average and policy methods performed somewhat better in our experiments.

The policy method:

$$\epsilon(s,a) = \epsilon_0 + \gamma^2 \sigma^2(s', \arg\max_{a' \in A_{s'}} \mu^T \phi(s',a')) \qquad (9)$$

The average method:

$$\epsilon(s,a) = \epsilon_0 + \gamma^2 \frac{\sum_{a' \in A_{s'}} \sigma^2(s',a')}{|A_{s'}|} \qquad (10)$$

The max method:

$$\epsilon(s,a) = \epsilon_0 + \gamma^2 \max_{a' \in A_{s'}} \sigma^2(s',a') \qquad (11)$$

The Boltzmann method:

$$\epsilon(s,a) = \epsilon_0 + \gamma^2 \frac{\sum_{a' \in A_{s'}} \sigma^2(s',a') e^{\frac{\mu^T \phi(s',a')}{\tau}}}{\sum_{a' \in A_{s'}} e^{\frac{\mu^T \phi(s',a')}{\tau}}} \qquad (12)$$

Regardless which heuristic method is used, the action selection can be made independently. For example, in our experiments in the next section we use Boltzmann action selection.

Kalman Filter Q-Learning belongs to the family of least squares value models, and therefore each update has complexity $O(n^2)$. This has traditionally given projected TD Learning methods an advantage over least squares methods. Although projected TD Learning is generally less efficient per iteration than least squares methods, each update only has complexity $O(n)$. This means that in practice, multiple updates to the projected TD Learning model can be made in the same time as a single update to the Kalman filter model.

2.2 APPROXIMATE KALMAN FILTER Q-LEARNING

We propose **Approximate Kalman Filter Q-Learning** (AKFQL) which updates the value model with only $O(n)$ complexity, the same complexity update as projected TD Learning. The only change in AKFQL is to simply ignore the off-diagonal elements in the covariance matrix. Only the variances on the diagonal of the covariance matrix are computed and stored. This approximation has linear update complexity, rather than quadratic, in the number of basis functions. Although KFQL outperforms AKFQL early on (on a per iteration basis), somewhat surprisingly, AKFQL overtakes it in the long run. We do not fully understand the mechanism of this improvement, but suspect that KFQL is overfitting.

Approximate Kalman Filter Q-Learning's calculations are simplified versions of KFQL's calculations which involve only the diagonal elements of the covariance matrix Σ. The Kalman gain can be computed by

$$\begin{aligned} d &= \sum_i {\phi_i}^2 \Sigma_{ii} + \epsilon \qquad (13) \\ G_i &= \frac{\Sigma_{ii} \phi_i}{d}. \qquad (14) \end{aligned}$$

The Kalman gain determines the relative impact the update makes on the elements of μ and Σ.

$$\mu_i \leftarrow \mu_i + G_i(\nu - \mu^T \phi) \tag{15}$$
$$\Sigma_{ii} \leftarrow (1 - G_i\phi_i)\Sigma_{ii} \tag{16}$$

The posterior values for μ and the diagonal of Σ can then be used as prior values in the next update. Finally, to compute the sample noise $\epsilon(s,a)$ we use $\sigma^2(s,a) = \sum_i \phi_i^2(s,a)\Sigma_{ii}$.

2.3 THE KFQL/AKFQL ALGORITHM

```
1 $\mu, \Sigma \leftarrow$ prior distribution over $r$
2 $s \leftarrow$ initial state
3 while policy generation in progress do
4     choose action $a \in A_s$
5     observe transition $s$ to $s'$
6     update $\mu$ and $\Sigma$ using $\nu(s,a)$ and $\epsilon(s,a)$
7     $s \leftarrow s'$
8     if isTerminal(s) then
9         $s \leftarrow$ initial state
```

Algorithm 1 : [Approximate] Kalman Filter Q-Learning

In Kalman Filter Q-Learning (KFQL) and Approximate Kalman Filter Q-Learning (AKFQL) we begin with our prior beliefs about the basis function weights r and we update them on each state transition, as shown in Algorithm 1.

3 EXPERIMENTAL RESULTS

Because our primary goal is policy generation rather than on-line performance, our techniques are not meant to balance exploration with exploitation. Instead of testing them on-line, we ran the algorithms off-line, periodically copied the current policy, and tested that policy. This resulted in a plot of the average control performance of the current policy as a function of the number of states visited by the algorithm. In each of our experiments, we ran each test $32 - 50$ times, and plotted the average of the results. At each measurement point of each test run, we averaged over several trials of the current policy.

In our results, we define a state as being **visited** when the MDP entered that state while being controlled by the algorithm. In the algorithms presented within this paper, each state visited also corresponds to one iteration of the algorithm. The purpose of these measurements is to show how the quality of the generated policy varies as the algorithm is given more time to execute. Thus, the efficiency of various algorithms can be compared across a range of running times.

3.1 CART-POLE

The **Cart-Pole** process is a well known problem in the MDP and control system literature (Anderson, 1986, Barto et al., 1983, Michie and Chambers, 1968, Schmidhuber, 1990, Si and Wang, 2001). The Cart-Pole process, is an instance of the **inverted pendulum** problem. In the Cart-Pole problem, the controller attempts to keep a long pole balanced atop a cart. At each time step, the controller applies a lateral force, F, to the cart. The pole responds according to a system of differential equations accounting for the mass of the cart and pole, the acceleration of gravity, the friction at the joint between the cart and pole, the friction between the cart and the track it moves on, and more.

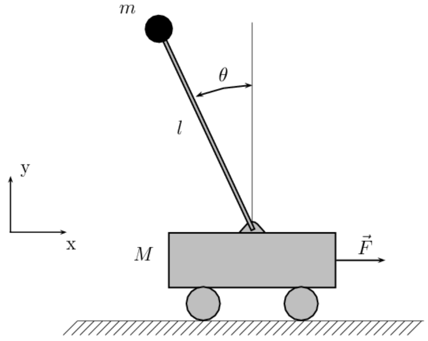


Figure 2: The Cart-Pole System

In our experiments, we used the corrected dynamics derived and described by Florian in (Florian, 2007).

In our experiments, we used a control interval of .1 seconds, and the values listed in table 2. At each control interval, the controller chooses a force to apply to the cart in $\{-5N, 0, 5N\}$. Then, this force and an additional force which is uniformly distributed in $[-2N, 2N]$ is also applied to the cart. The controller receives a reward of 1 at each time step except when a terminal state is reached. The problem is undiscounted with $\gamma = 1$. Thus, the number of simulation steps before the pole falls over is equal to the total reward received.

3.1.1 Basis Functions

For the Cart-Pole process, we used a simple bilinear interpolating kernel. For each action, a grid of points is defined in the state space (θ, ω). Each point corresponds to a basis function. Thus, every state-action pair corresponds to a point within a grid box with a basis function point at each of the four corners. All

Table 2: Variable definitions for the Cart-Pole process

VARIABLE	**DEFINITION**
$m_c = 8.0kg$	Mass of the cart
$m_p = 2.0kg$	Mass of the pole
$l = .5m$	Length of the pole
$g = 9.81m/s^2$	Acceleration of gravity
$\mu_c = .001$	Coefficient of friction between the cart and the track
$\mu_p = .002$	Coefficient of friction between the pole and the cart
θ	Angle of the pole from vertical (radians)
$\dot{\theta} = \omega$	Angular velocity of the pole (rad/s)
$\ddot{\theta}$	Angular acceleration of the pole (rad/s^2)
F	Force applied to the cart by the controller (Newtons)

basis function values for a particular state are zero except for these four functions. The value of each of the non-zero basis functions is defined by:

$$\tilde{\phi_i} = \left(1 - \left|\frac{\theta - \theta_{\phi_i}}{s_\theta}\right|\right)\left(1 - \left|\frac{\omega - \omega_{\phi_i}}{s_\omega}\right|\right) \quad (17)$$

$$\phi_i = \frac{\tilde{\phi_i}}{\sum_i \tilde{\phi_i}} \quad (18)$$

where $s_\theta = \frac{\pi}{2}$ and $s_\omega = \frac{1}{4}$, and the basis function points are defined for each combination of $\theta_{\phi_i} \in \{-\pi, -\frac{\pi}{2}, 0, \frac{\pi}{2}, \pi\}$ and $\omega_{\phi_i} \in \{-\frac{1}{2}, -\frac{1}{4}, 0, \frac{1}{2}, \frac{1}{4}\}$. This set of basis functions yields $5 \times 5 = 25$ basis functions for each of the three possible actions, for a total of 75 basis functions.

Each test on Cart Pole consisted of averaging over 50 separate runs of the generation algorithm, evaluating each run at each test point once. Evaluation runs were terminated after 72000 timesteps, corresponding to two hours of simulated balancing time. Therefore, the best possible performance is 72000.

For projected TD-Learning, a learning rate of $.5\frac{1e6}{1e6+t}$ was used. This learning rate was determined by testing every combination of $s \in \{.001, .005, .01, .05, .1, .5, 1\}$ and $c \in \{1, 10, 100, 1000, 10000, 100000, 1000000, 10000000\}$ for the learning rate, $\alpha_n = s\frac{c}{c+n}$. For KFQL and AKFQL, a sensor variance of $\sigma = .1$, a prior variance of $\Sigma_{ii} = 10000$.

3.2 CASHIER'S NIGHTMARE

The **Cashier's Nightmare**, also called **Klimov's problem** is an extension of the multi-armed bandit problem into a queueing problem with a particular structure. In the Cashier's Nightmare, there are d queues and k jobs. At each time step, each queue, i, incurs a cost proportional to it's length, $g_i x_i$. Thus, the total immediate reward at each time step is $-g^T x_t$. Further, at each time step the controller chooses a queue to service. When a queue is serviced, its length is reduced by one, and the job then joins a new queue, j, with probabilities, p_{ij}, based on the queue serviced. Consequently, the state space, x, is defined by the lengths of each queue with x_i equaling the length of queue i in state x.

The Cashier's Nightmare has a state space of $\left\langle {d \atop k} \right\rangle = \binom{d+k-1}{k}$ states: one for each possible distribution of the k jobs in the d queues. The instance of the Cashier's Nightmare that we used has $d = 100$ queues, and $k = 200$ jobs, yielding a state space of $\left\langle {100 \atop 200} \right\rangle \approx 1.39 \times 10^{81}$ states. This also means that each timestep presents 100 possible actions and each state transition, given an action, presents 100 possible state transitions. In this instance, we set $g_i = \frac{i}{d}$ with $i \in \{1, 2, ..., d\}$ and randomly and uniformly selected probability distributions for p_{ij}.

It is worth noting that although we use this problem as a benchmark, the optimal controller can be derived in closed form (Buyukkoc et al., 1983). However, this is a challenging problem with a large state space, and therefore presents a good benchmark for approximate dynamic programming techniques.

3.2.1 Basis Functions

The basis functions that we use on Cashier's Nightmare are identical to those used by Choi and Van Roy in (Choi and Van Roy, 2006). One basis function is defined for each queue, and it's value is based on the length of the queue and the state transition probabilities:

$$\phi_i(x, a) = x_i + (1 - \delta_{x_i,0})(p_{a,i} - \delta_{a,i}) \quad (19)$$

This way, at least the short term effect of each action is reflected in the basis functions.

For projected TD-Learning, a learning rate of $.1\frac{1e3}{1e3+t}$ was used. This learning rate was determined by testing every combination of $s \in \{.001, .01, .1, 1\}$ and $c \in \{1, 10, 100, 1000, 10000\}$ for the learning rate, $\alpha_n = s\frac{c}{c+n}$. For KFQL and AKFQL, a sensor variance of $\sigma = 1$, a prior variance of $\Sigma_{ii} = 20$.

3.3 CAR-HILL

The **Car-Hill** control problem, also called the **Mountain-Car** problem is a well-known continuous state space control problem in which the controller attempts to move an underpowered car to the top of a mountain by increasing the car's energy over time. There are several variations of the dynamics of the Car-Hill problem; the variation we use here is the same as that found in (Ernst et al., 2005).

A positive reward was given when the car reached the summit of the hill with a low speed, and a negative reward was given when it ventured too far from the summit, or reached too high of a speed:

$$r(p, v) = \begin{cases} -1 & \text{if } p < -1 \text{ or } |v| > 3 \\ 1 & \text{if } p > 1 \text{ and } |v| \leq 3 \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

The implementation we used approximates these differential equations using Euler's method with a timestep of .001s. The control interval used was .1s, and the controller's possible actions were to apply forces of either $+4N$ or $-4N$. A decay rate of $\gamma = .999$ was used.

Each test on Car-Hill consisted of averaging over 32 separate runs of the generation algorithm, evaluating each run at each test point once. Evaluation runs were terminated after 1000 timesteps, ignoring the small remaining discounted reward.

3.3.1 Basis Functions

We used bilinearly interpolated basis functions with an 8x8 grid spread evenly over the $p \in [-1, 1]$ and $v \in [-3, 3]$ state space for each action. This configuration yields $8x8x2 = 128$ basis functions. The prior we used was set to provide higher prior estimates for states with positions closer to the summit:

$$\mu_0(p, v) = \max\left\{0, 1 - \frac{2(1-p)}{3}\right\} \tag{21}$$

For projected TD-Learning, a learning rate of $.1\frac{1e3}{1e3+t}$ was used. This learning rate was determined by testing every combination of $s \in \{.001, .01, .1, 1\}$ and $c \in \{1, 10, 100, 1000, 10000\}$ for the learning rate, $\alpha_n = s\frac{c}{c+n}$. For KFQL and AKFQL, a sensor variance of $\sigma = .5$, a prior variance of $\Sigma_{ii} = .1$.

3.4 RESULTS

Kalman Filter Q-Learning has mixed performance relative to a well-tuned projected TD Learning implementation. For the Cart-Pole process (Figure 3), KFQL provides between 100 and 5000 times the efficiency of PTD. However, on the Cashier's Nightmare process (Figure 4) and the Car Hill process (Figure 5), KFQL was plagued by numerical instability, apparently because KFQL underestimates the variance leading to slow policy changes. Increasing the constant sensor noise, ϵ_0, can reduce this problem somewhat, but may not eliminate the issue.

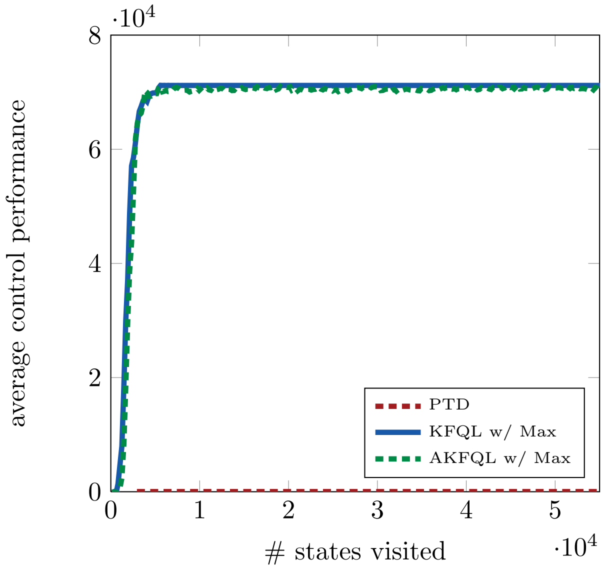


Figure 3: PTD, KFQL, and AKFQL on Cart-Pole

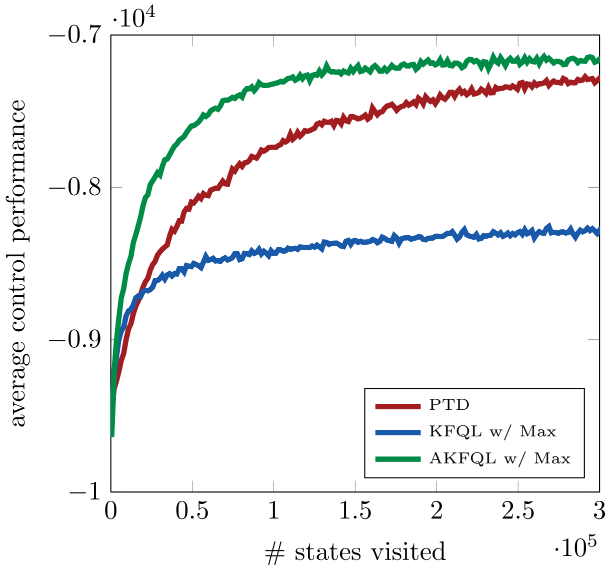


Figure 4: PTD, KFQL, and AKFQL on Cashier's Nightmare

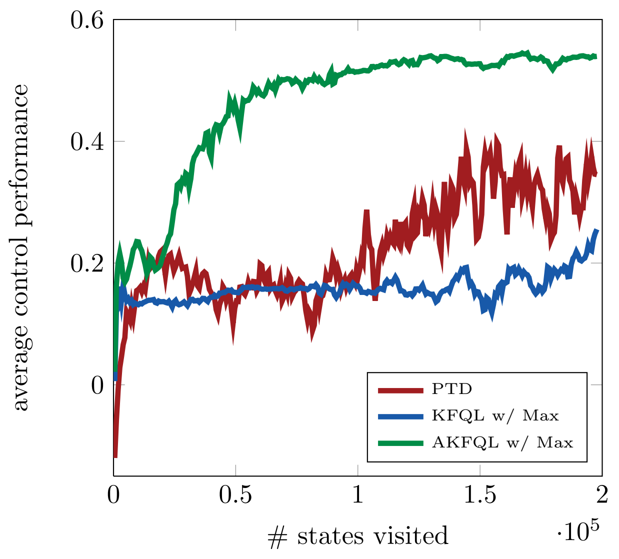


Figure 5: PTD, KFQL, and AKFQL on Car-Hill

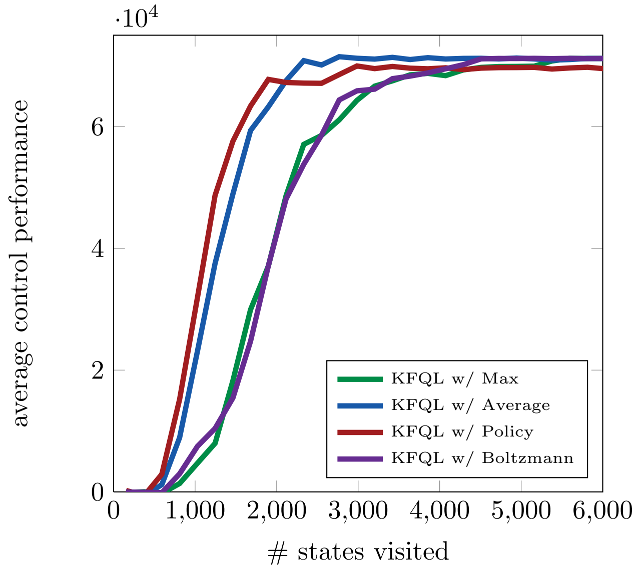


Figure 6: Comparison of sensor noise models when using KFQL on Cart-Pole

Approximate Kalman Filter Q-Learning, on the other hand, provides a very large benefit over PTD on all of the tested applications (Figures 3, 4, and 5). Interestingly, in two of the experiments, AKFQL also outperforms KFQL by a large margin, even though the effort per iteration is much less for AKFQL. This is likely due to the fact that AKFQL ignores any dependencies among basis functions. Instead of solving the full least-squares problem (that a Kalman filter solves), AKFQL is performing an update somewhere between the least-squares problem and the gradient-descent update of TD-Learning. This middle-ground provides the benefits of tracking the variances on each basis function's weight, and the freedom from the dependencies and numerical stability issues inherent in KFQL. Additionally, AKFQL's update complexity is linear, just as is PTD's, so it truly outperforms projected TD-Learning in our experiments.

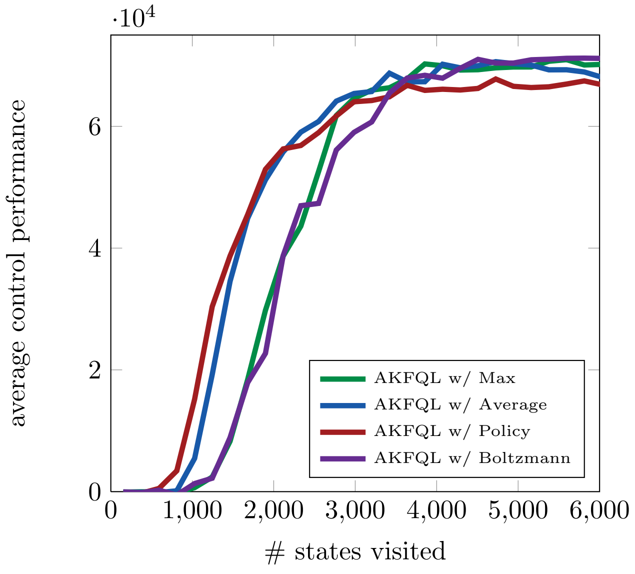


Figure 7: Comparison of sensor noise models when using AKFQL on Cart-Pole

The choice of signal noise heuristic has a small effect for the Cart-Pole process under either KFQL (Figure 6) or AKFQL (Figure 7). The average and policy methods appear to work better than the max and Boltzmann methods. The other figures in this section were generated using the max method.

4 RELATED WORK

Kalman Filter Q-Learning uses a projected value model, where a set of basis functions $\phi(s, a)$ project the MDP state action space onto the basis space. In these methods, basis function weights r are used to estimate Q-values, $Q(s, a) \approx r^T \phi$, in order to develop high performance policies.

4.1 PROJECTED TD LEARNING

Projected TD Learning (Roy, 1998, Sutton, 1988, Tsitsiklis and Roy, 1996) is a popular projection-based method for approximate dynamic programming. The version of TD Learning that we consider here is the off-policy Q-learning variant with $\lambda = 0$. Thus, our objective is to find an r such that $Q^*(s,a) \approx r^T\phi(s,a)$ because we are primarily concerned with finding efficient off-line methods for approximate dynamic programming.

The projected TD Learning update equation is:

$$\begin{aligned} r_{t+1} = r_t + \alpha_t\phi(s_t,a_t)\times \\ \big[(F\Phi r_t)(s_t,a_t) + w_{t+1} - (\Phi r_t)(s_t,a_t)\big] \end{aligned} \tag{22}$$

where F is a weighting matrix and Φ is a matrix of basis function values for multiple states. We use stochastic samples to approximate $(F\Phi r_t)(x_t)$. The standard Q-Learning variant that we consider here sets:

$$\begin{aligned} (F\Phi r_t)(s_t,a_t) + w_{t+1} = \\ R(s_t,a_t,s_{t+1}) + \gamma \max_{a\in A_{s_{t+1}}} r^T\phi(s_{t+1},a) \end{aligned} \tag{23}$$

where the state transition $(s_t,a_t) \rightarrow (s_{t+1})$ is sampled according to the state transition probabilities, $P(s_t \rightarrow s_{t+1}|s_t,a_t)$, of the MDP. Substituting Equation 23 into Equation 22, we arrive at the update equation for this variant of projected TD Learning:

$$\begin{aligned} r_{t+1} = r_t + \alpha_t\phi(s_t,a_t)\times \\ \Big[\big(R(s_t,a_t,s_{t+1}) + \gamma \max_{a\in A_{s_{t+1}}} r^T\phi(s_{t+1},a)\big) \\ -(\Phi r_t)(s_t,a_t)\Big] \end{aligned} \tag{24}$$

The convergence properties of this and other versions of TD learning have been studied in detail (Dayan, 1992, Forster and Warmuth, 2003, Pineda, 1997, Roy, 1998, Schapire and Warmuth, 1996, Sutton, 1988, Tsitsiklis and Roy, 1996). Unlike our Kalman filter methods, PTD has no sense of how well it knows a particular basis functions weight, and adjusts the weights on all basis functions at the global learning rate. As with the tabular value model version of TD learning, the choice of learning rates may greatly impact the performance of the algorithm.

4.2 LEAST SQUARES POLICY ITERATION

Least squares policy iteration (LSPI) (Lagoudakis et al., 2003) is an approximate policy iteration method which at each iteration samples from the process using the current policy, and then solves the least-squares problem:

$$\text{minimize} \quad ||\hat{A}_t w - \hat{b}_t||_\mu \tag{25}$$

where $\hat{A}$ and $\hat{b}$ are:

$$\begin{aligned} \hat{A} &= \frac{1}{L}\sum_{i=1..L} \phi(s_i,a_i)\big[\phi(s_i,a_i) - \gamma\phi(s'_i,\pi(s'_i))\big]^T \\ &\approx \Phi^T\Delta_\mu(\Phi - \gamma P\Pi\Phi) \end{aligned} \tag{26}$$

$$\begin{aligned} \hat{b}_{t+1} &= \hat{b}_t + \phi(s_t,a_t)r_t \\ &\approx \Phi^T\Delta_\mu R \end{aligned} \tag{27}$$

where P, R, Π, and Φ are matrices of probabilities, rewards, policies, and basis function values, respectively.

Solving this least squares problem is equivalent to using KFQL with a fixed-point update method, an infinite prior variance and infinite dynamic noise. LSPI is effective for solving many MDPs, but it can be prone to policy oscillations. This is due to two factors. First, the policy biases $\hat{A}$ and $\hat{b}$, can result in a substantially different policy at the next iteration, which could lead to a ping-ponging effect between two or more policies. Second, information about $\hat{A}$ and $\hat{b}$ is discarded between iterations, providing no dampening to the oscillations and jitter caused by simulation noise and the sampling bias induced by the policy.

4.3 THE FIXED POINT KALMAN FILTER

The Kalman filter has also been adapted for use as a value model in approximate dynamic programming by Choi and Van Roy (Choi and Van Roy, 2006). **Fixed point Kalman filter** models the same multivariate normal weights r as KFQL but parametrizes them with means r_t and precision matrix ${H_t}^{-1}$ such that:

$$\begin{aligned} r_{t+1} = r_t + \tfrac{1}{t}H_t\phi(s_t,a_t)\times \\ \big[(F\Phi r_t)(s_t,a_t) + w_{t+1} - (\Phi r_t)(s_t,a_t)\big] \end{aligned} \tag{28}$$

where $(F\Phi r_t)(s_t,a_t) + w_{t+1}$ is the same as in Equation 23, yielding the update rule:

$$\begin{aligned} r_{t+1} = r_t + \tfrac{1}{t}H_t\phi(s_t,a_t)\times \\ \Big[\big(R(s_t,a_t,s_{t+1}) + \gamma \max_{a\in A_{s_{t+1}}} r^T\phi(s_{t+1},a)\big)- \\ (\Phi r_t)(s_t,a_t)\Big] \end{aligned} \tag{29}$$

where H_t is defined as:

$$H_t = \left[\frac{1}{t}\sum_{i=1}^{t}\phi(s_i,a_i)\phi^T(s_i,a_i)\right]^{-1} \qquad (30)$$

Non-singularity of H_t can be dealt with using any known method, although using the psudo-inverse is standard. The choice of an initial r_0, H_0 and the inversion method for H are similar to choosing the prior means and variance over r in the KFQL. The fixed point Kalman filter represents the covariance of r by a precision matrix rather than the covariance matrix used by KFQL. This difference is significant because KFQL performs no matrix inversion and avoids many non-singularity and numerical stability issues.

4.4 APPROXIMATE KALMAN FILTERS

Many methods for approximating and adapting the Kalman filter have been developed for other applications (Chen, 1993). Many of these methods can be used in this application as well. AKFQL is just one simple approximation technique that works well and achieves $O(n)$ instead of $O(n^2)$ complexity updates. Other possible approximation techniques include fixed-rank approximation of Σ and particle filtering (Doucet et al., 2001, Gordon et al., 1993, Kitagawa and Gersch, 1996, Rubin, 1987).

5 CONCLUSIONS

In this paper we have presented two new methods for policy generation in MDPs with continuous state spaces. The Approximate Kalman Filter Q-Learning algorithm provides significant improvement on several benchmark problems over existing methods, such as projected TD-Learning, as well as our other new method, Kalman Filter Q-Learning. Continuous state MDPs are challenging problems that arise in multiple areas of artificial intelligence, and we believe that AKFQL provides a significant improvement over the state of the art. We believe these same benefits apply to other problems where basis functions are useful, such as MDPs with large discrete sample spaces and mixed continuous and discrete state spaces.

There are a variety of ways this work could be extended. The Kalman filter allows for linear dynamics in the distribution of basis function weights r during transitions from one MDP state action pair to another or from one policy to another. Another possibility is to incorporate more of a fixed point approach, recognizing the dependence between the prediction variance $\sigma^2(s,a)$ and the signal noise $\epsilon(s,a)$ conditioned on r. Still another possibility is to use an approximation, ignoring off-diagonal elements like AKFQL, in the traditional fixed point approach. Finally, we could apply a similar Kalman filter model to perform policy iteration directly rather than by Q-Learning.

Acknowledgements

We thank the anonymous referees for their helpful suggestions.

References

Charles W. Anderson. *Learning and Problem Solving with Multilayer Connectionist Systems.* PhD thesis, University of Massachusetts, Department of Computer and Information Science, Amherst, MA, USA, 1986.

A.G. Barto, R.S Sutton, and C. Anderson. Neuron-like adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13:834–846, 1983.

Richard Bellman. A markovian decision process. *Indiana Univ. Math. J.*, 6:679–684, 1957. ISSN 0022-2518.

Dimitri P. Bertsekas. Distributed dynamic programming. *IEEE Transactions on Automated Control*, AC-27:610–616, 1982.

C. Buyukkoc, Pravin Varaiya, and Jean Walrand. Extensions of the multi-armed bandit problem. Technical Report UCB/ERL M83/14, EECS Department, University of California, Berkeley, 1983.

Q. Chen. *Approximate Kalman Filtering.* Series in Approximations and Decompositions. World Scientific, 1993. ISBN 9789810213596.

David Choi and Benjamin Van Roy. A generalized kalman filter for fixed point approximation and efficient temporal-difference learning. *Discrete Event Dynamic Systems*, 16(2):207–239, 2006.

Peter Dayan. The convergence of td(λ) for general λ. *Machine Learning*, 8:341–362, 1992.

A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice.* Statistics for Engineering and Information Science. Springer, 2001. ISBN 9780387951461.

Damien Ernst, Pierre Geurts, Louis Wehenkel, and L. Littman. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6: 503–556, 2005.

Răzvan V. Florian. Correct equations for the dynamics of the cart-pole, 2007.

Jürgen Forster and Manfred K. Warmuth. Relative loss bounds for temporal-difference learning. *Machine Learning*, 51(1):23–50, 2003.

N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107 –113, apr 1993. ISSN 0956-375X.

Ronald A. Howard. *Dynamic Programming and Markov Processes*. The M.I.T. Press, Cambridge, MA, USA, 1960. ISBN 0-262-08009-5.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D): 35–45, 1960.

G. Kitagawa and W. Gersch. *Smoothness Priors Analysis of Time Series*. Lecture Notes in Statistics. Springer, 1996. ISBN 9780387948195.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009. ISBN 978-0-262-01319-2.

Michail G. Lagoudakis, Ronald Parr, and L. Bartlett. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:2003, 2003.

D. Michie and R.A. Chambers. BOXES: An experiment in adaptive control. In E. Dale and D. Michie, editors, *Machine Intelligence 2*, pages 137–152. Oliver and Boyd, Edinburgh, 1968.

Fernando J. Pineda. Mean-field theory for batched td (&lgr;). *Neural Comput.*, 9(7):1403–1419, October 1997. ISSN 0899-7667.

Ben Van Roy. *Learning and Value Function Approximation in Complex Decision Processes*. PhD thesis, MIT, 1998.

Donald B. Rubin. The calculation of posterior distributions by data augmentation: Comment: A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The sir algorithm. *Journal of the American Statistical Association*, 82(398):pp. 543–546, 1987. ISSN 01621459.

Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, second edition, 2003. ISBN 978-81-203-2382-7.

Robert E. Schapire and Manfred K. Warmuth. On the worst-case analysis of temporal-difference learning algorithms. *Machine Learning*, 22(1-3):95–121, 1996.

Jrgen Schmidhuber. Networks adjusting networks. In *Proceedings of 'Distributed Adaptive Neural Information Processing', St.Augustin*, pages 24–25. Oldenbourg, 1990.

J. Si and Yu-Tsung Wang. On-line learning control by association and reinforcement. *Neural Networks, IEEE Transactions on*, 12(2):264 –276, mar 2001. ISSN 1045-9227. doi: 10.1109/72.914523.

Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988. ISSN 0885-6125. 10.1007/BF00115009.

John N. Tsitsiklis and Benjamin Van Roy. Analysis of temporal-diffference learning with function approximation. In *NIPS*, pages 1075–1081, 1996.

C. Watkins. *Learning from Delayed Rewards*. PhD thesis, University of Cambridge, England, 1989.

Greg Welch and Gary Bishop. An introduction to the kalman filter. Technical report, SIGGRAPH 2001, 2001. Course Notes.

Finite-Time Analysis of Kernelised Contextual Bandits

Michal Valko, Nathan Korda, Rémi Munos
SequeL team
INRIA Lille - Nord Europe, France

Ilias Flaounas, Nello Cristianini
Intelligent Systems Laboratory
University of Bristol, UK

Abstract

We tackle the problem of online reward maximisation over a large finite set of actions described by their contexts. We focus on the case when the number of actions is too big to sample all of them even once. However we assume that we have access to the similarities between actions' contexts and that the expected reward is an *arbitrary* linear function of the contexts' images in the related reproducing kernel Hilbert space (RKHS). We propose KernelUCB, a kernelised UCB algorithm, and give a cumulative regret bound through a frequentist analysis. For contextual bandits, the related algorithm GP-UCB turns out to be a special case of our algorithm, and our finite-time analysis improves the regret bound of GP-UCB for the agnostic case, both in the terms of the kernel-dependent quantity and the RKHS norm of the reward function. Moreover, for the linear kernel, our regret bound matches the lower bound for contextual linear bandits.

1 Introduction

There are many situations in which an environment repeatedly provides an agent with a very large number of actions together with some contextual information (Cesa-Bianchi & Lugosi, 2006). These actions yield rewards when chosen and the agent wants to continually choose actions that yield high expected reward while not having enough time to explore them all. Thus it is natural to learn a relationship between the context provided for each action and the expected reward it produces. Kernel methods (Shawe-Taylor & Cristianini, 2004) provide a way to extract from observations possibly non-linear relationships between the contexts and the rewards while only using similarity information between contexts. In many applications similarity information is cheaply computable. In some situations the contexts are not even available and instead only similarities are given (Chen, Garcia, Gupta, Rahimi, & Cazzanti, 2009).

A typical example (Li, Chu, Langford, & Schapire, 2010), is the case of online advertisement in which one needs to continually show the most relevant ads to users viewing a website; since there is a simple binary reward of 1 for a click on the ad shown and 0 otherwise it is always costly to show ads that have only a small chance of being clicked on. Another example is a recommender system for relevant content from a large number of available news feeds (Steinberger, Pouliquen, & Van der Goot, 2009); here it is assumed that we can assess the relevance of the content of a feed based on information such as the anchor text of the feed link without having to get and process the actual feed content, which is a costly operation.

Our modelling assumption is that the expected reward obtained from choosing an action is a function of the features associated with that action. In the advertisement example the features are built from webpage content and user attributes. In the news feeds, the features come from easily retrievable information such as URLs, feed titles, or anchor text. We refer to the features as *contexts* and to the resulting problems of maximising cumulative reward as *contextual bandit problems.* One aspect that makes this setting different from related settings is the possibly changing decision set.

Previous approaches (Li et al., 2010; Chu, Li, Reyzin, & Schapire, 2011; Auer, 2002) to contextual bandit problems have often assumed that the functional relationship between the features and the expected rewards is linear. However the availability of similarity information gives us the opportunity to search for a linear relationship in a reproducing kernel Hilbert space (RKHS) defined by these similarities to discover a non-linear relationship between the context and the reward. Recently, Srinivas, Krause, Kakade, and Seeger

(2010) proposed the GP-UCB algorithm that optimises a function θ^* sampled from a Gaussian Process (GP) prior. In this paper we take an agnostic approach (Table 1) and provide the KernelUCB algorithm which comes directly from kernelising contextual linear bandits. KernelUCB is a kernel-based upper confidence bound algorithm which, given the similarity between two data points, uses the dualisation of regularised linear regression in the RKHS to find upper confidence bounds on the expected rewards of each action, and then chooses an action with the highest upper confidence bound. When the kernel is just the dot product between feature vectors KernelUCB is identical to LinUCB (Li et al., 2010), i.e., KernelUCB is a non-linear extension of LinUCB.

Our main contribution is a theoretical analysis of this approach. While kernelisation of linear bandits is straighforward, the analysis has to deal with an RKHS with potentially infinite dimension. We provide a data-dependent performance bound based on a notion of the effective dimension $\tilde{d}$. This quantity roughly measures the number of directions in the RKHS along which the data mostly lies. We are able to provide a cumulative regret bound that scales as $\tilde{O}(\sqrt{T\tilde{d}})$, where T is the time and $\tilde{O}$ hides log factors. When the kernel is just the dot product between contexts, $\tilde{d}$ is upper bounded by the dimension of the contexts, and we recover the regret bounds for LinUCB for contextual linear bandits as a special case. The GP-UCB algorithm is also a special case of KernelUCB when the regulariser is set to the model noise, and we make (Section 4.1) a clear comparison with the *agnostic* analysis of GP-UCB (Srinivas et al., 2010), i.e. when their reward function θ^* is not sampled from a GP. For this agnostic case, Srinivas et al. (2010) obtain a cumulative regret bound $\tilde{O}(I(y_T;\theta^*)\sqrt{T})$ where $I(y_T;\theta^*)$ is the information gain between θ^* and the observed samples y_T. We show that $I(y_T;\theta^*)$ is $\Omega(\tilde{d})$ and since our bound only scales with $\sqrt{\tilde{d}}$, our analysis matches the lowerbound for the linear case, unlike the agnostic analysis of GP-UCB. Furthermore, due to the link between $\tilde{d}$ and $I(y_T;\theta^*)$ we can provide the *data-independent* worst case upperbounds for the popular kernels (such as RBF) by plugging the upperbounds $I(y_T;\theta^*)$ derived by Srinivas et al. (2010) into our improved analysis. Our analysis also gives us a guideline on how to set the regularisation parameter.

Section 2 presents the basic linear contextual bandit model and related work. In Section 3 we derive the KernelUCB algorithm by directly kernelising contextual linear bandits. In Section 4 we analyse KernelUCB, provide an upper bound on the cumulative regret and describe the tradeoff between the regularization and the RHKS norm of the reward function.

	Bayesian	**Frequentist**
regression	GP-Regression	Kernel Ridge Regression
bandits	GP-UCB	**KernelUCB** this paper

Table 1: Bayesian and frequentist approaches to kernelized regression and contextual bandits

2 Background

2.1 Basic Model

We describe the basic settings and goals of linear contextual bandit problems. At each time t, for each action $a \in \mathcal{A} := \{1, \dots, N\}$, there is an associated context vector $x_{a,t} \in \mathbb{R}^d$. If action a is chosen at time t we have $a_t = a$ and receive a reward $r_{a,t}$ drawn from a distribution $\nu_{a,x_{a,t}}$. An algorithm π is a method for choosing an action at time t given the history i.e., the previously observed contexts, actions and rewards, and the current context:

$$H_{t-1} := \left(\{x_{a,j}\}_{a\in\mathcal{A}}, a_j, r_{a_j,j}\right)_{j<t} \cup \{x_{a,t}\}_{a\in\mathcal{A}},$$
$$\pi : H_{t-1} \mapsto \pi_t \in \mathcal{P}(\mathcal{A}),$$

where $\mathcal{P}(\mathcal{A})$ denotes the set of probability distributions over $\mathcal{A}$. For simplicity, we define $x_t := x_{a_t,t}$ and $r_t := r_{a_t,t}$ to be the context and the reward at the time t.

In the case of classical bandits, the reward distributions $\nu_{a,x_{a,t}}$ are independent of the context vectors, $x_{a,t}$. In this case we define the optimal action as $a^* := \arg\max_{a\in\mathcal{A}}\{\mathbb{E}(r_a)\}$ and define the regret of an algorithm at time T to be:

$$R(T) := \sum_{t=1}^{T} r_{a^*,t} - r_t.$$

For linear contextual bandits we assume a linear relationship between contexts and mean rewards,

$$\mathbb{E}[r_{a,t} \mid x_{a,t}] = x_{a,t}^{\mathsf{T}}\theta^*,$$

for some fixed but unknown vector $\theta^* \in \mathbb{R}^d$. Note, that θ^* is the same for all actions and thus this problem is also called a fixed design setting (Bubeck & Cesa-Bianchi, 2012). In some (noncontextual) linear bandit settings (Dani, Hayes, & Kakade, 2008), the contexts do not change and $x_{a,t} = x_a$.

In this paper we consider the case when the contexts, and subsequently the optimal action, *can change* over time. Thus we have $a_t^* := \arg\max_{a\in\mathcal{A}}\{\mathbb{E}(r_{a,t} \mid x_{a,t})\}$

and the (contextual) regret of an algorithm at time T becomes:

$$R(T) := \sum_{t=1}^{T} r_{a_t^*,t} - r_t. \qquad (1)$$

The aim in both of these situations is to find an algorithm which minimises the regret at time T.

2.2 UCB Algorithms

Upper confidence bound (UCB) algorithms (Lai & Robbins, 1985) provide a simple but efficient heuristic approach to bandit problems. The central idea is to maintain for each action, a, an estimate of the mean reward $\hat{\mu}_{a,t}$ and a confidence interval around that mean with width $\hat{\sigma}_{a,t}$. At each time t the algorithm then chooses the action with the highest upper confidence bound $\hat{\mu}_{a_t} + \hat{\sigma}_{a,t}$; thus an action a is selected if either it has a high estimated mean, or if there is much uncertainty about the action so that the width $\hat{\sigma}_{a,t}$ is large.

For classical bandits (with no contextual information) it is possible to obtain finite time analyses of such algorithms along the following lines: Construct the widths $\hat{\sigma}_{a,t}$ so that they are large when a has not been played often but small when it has been played a large number of times already, for example by relating them to the standard deviations of the estimates $\hat{\mu}_{a,t}$. Assume that a suboptimal action, a, has been played a large number of times. Then through tools such as the Azuma-Hoeffding inequality one can expect to obtain high probability bounds on the events that $\hat{\mu}_{a,t}$ is close to μ_a. From the construction of the widths it follows that $\hat{\mu}_{a,t} + \hat{\sigma}_{a,t}$ will also be close to μ_a. In this way as soon as a sub-optimal action a has been played enough times so that $\hat{\mu}_{a,t} + \hat{\sigma}_{a,t} < \hat{\mu}_{a^*}$, the probability this action will be played again becomes very small. Such analyses typically conclude that UCB algorithms are close to optimal, and they motivate the choice of widths relating to the standard deviations of the estimates $\hat{\mu}_{a,t}$.

2.3 UCBs for Linear Contextual Bandits

Since we assume that there is a functional relationship between the expected rewards of an action and the feature vectors observed, constructing the estimates $\hat{\mu}_{a,t}$ and the widths $\hat{\sigma}_{a,t}$ can be approached by regression. In particular, when we assume a linear model we can use regularised least squares regression to estimate the mean rewards:

$$\hat{\mu}_{a,t} := x_{a,t}^{\intercal}\hat{\theta}_t$$

where $\hat{\theta}_t := C_t^{-1}X_t^{\intercal}y_t, y_t := \{r_{a_1,1}, \ldots, r_{a_t,t}\}^{\intercal}$, $X_t := \{x_{a_1,1}, \ldots, x_{a_t,t}\}^{\intercal}$, and $C_t := X_t^{\intercal}X_t + \gamma I_d$ for some $\gamma > 0$. Appropriate widths for the confidence intervals can be described in terms of the Mahalanobis distance of $x_{a,t}$ from the centre of mass of X_t:

$$\hat{\sigma}_{a,t} = \sqrt{x_{a,t}^{\intercal}C_t^{-1}x_{a,t}}$$

These widths relate to variance in the data: For instance in the case of standard normal noise (i.e. when the rewards satisfy $r_{a,t} = x_{a,t}^T\theta^* + \varepsilon_{a,t}$, where all $\varepsilon_{a,t} \sim \mathcal{N}(0,1)$), $\hat{\sigma}_{a,t}^2$ is exactly the variance of $\hat{\mu}_{a,t}$. Even when no assumption is made on the noise, this Mahalanobis distance has the property of being small when $x_{a,t}$ is close to the center of mass of data X_t, and large otherwise. Consequently a generic UCB type algorithm based on the estimators $\hat{\mu}_{a,t}$ and $\hat{\sigma}_{a,t}^2$ chooses an action a_t at time t such that:

$$a_t = \arg\max_{a \in A} \left(x_{a,t}^{\intercal}C_t^{-1}X_t^{\intercal}y_t + \eta\sqrt{x_{a,t}^{\intercal}C_t^{-1}x_{a,t}} \right),$$

where $\eta = \eta(t)$ is some (possibly time dependent) deterministic parameter of the algorithm which we call the *exploration* parameter.

Based on these ideas, Li et al. (2010) propose LinUCB which treats $\eta(t) = \eta$ as a constant that needs to be optimised. While this algorithm is simple to understand and implement in practice, no optimal theoretical regret analysis exists in the literature for LinUCB. Instead Chu et al. (Chu et al., 2011) give a theoretical analysis of a related algorithm, SupLinUCB, and achieve with probability $1 - \delta$ a regret bound of:

$$O\left(\sqrt{Td\ln^3(NT\ln(T)/\delta)}\right).$$

2.4 Related Work

The most related work to our setting is LinUCB (Li et al., 2010) and SupLinUCB (Chu et al., 2011), which were inspired by SupLinRel (Auer, 2002), an early algorithm for linear contextual bandits. Instead of using regularised linear regression SupLinRel uses eigendecomposition to make a pseudo-inverse of the covariance matrix. A discussion of practical advantages of SupLinUCB over SupLinRel can be found in (Li et al., 2010). SupLinRel achieves a regret bound $O((Td\ln^{3/2}(2NT\ln(T)/\delta))^{1/2})$.

Interestingly, one can derive an instantiation of KernelUCB in the Bayesian setting. This is the case of GP-UCB (Srinivas et al., 2010) a special case of KernelUCB, which assumes that the reward function is drawn from a GP prior. The conceptual difference between the KernelUCB and GP-UCB is similar to the difference between kernel regression and GP-regression (Table 1). Nevertheless, Srinivas et al. (2010) also provide a frequentist analysis of GP-UCB, which we compare to in Section 4.1. Krause and Ong (2011) later propose CGP-UCB, an extension of GP-UCB for the setting when each action has its own intrinsic features, as well as features associated to its changing environment. It therefore uses possibly different kernels for the action and the context spaces.

Slivkins (2009) takes advantage of similarity information between contexts, where he builds on previous work (Kleinberg, Slivkins, & Upfal, 2008; Lu, Pál, & Pál, 2010) that assume only a metric space structure on the context and action spaces. The setting in (Slivkins, 2009) is different from ours: they assume a Lipschitz property in a similarity space, which is a weaker condition than in our setting, but as a consequence their bound depends more heavily on the relevant dimensions (the covering dimensions of the context and action spaces appears in the exponent of T whereas our effective dimension appears as a multiplicative factor only).

Another well known related family are the Confidence-Ball algorithms (Abbasi-Yadkori, Pal, & Szepesvari, 2011; Dani et al., 2008; Rusmevichientong & Tsitsiklis, 2010). These solve the linear bandit problem in which the action space is the context space and there is a reward linear in contexts. When we fix the contexts in our own setting we recover the linear bandit model for a finite action set and in that sense our setting is more general. For continuous action space linear bandits the attainable lower bound for the regret is $\Omega(d\sqrt{T})$ (Dani et al., 2008), whereas for finite action space linear contextual bandits the attainable lower bound on regret is $\Omega(\sqrt{dT})$ (Chu et al., 2011).

A set of algorithms based on EXP4 (Auer, Cesa-Bianchi, Freund, & Schapire, 2003) such as EXP4.P (Beygelzimer, Langford, Li, Reyzin, & Schapire, 2010) or Policy Elimination (Dudik, Hsu, Kale, Karampatziakis, Langford, Reyzin, & Zhang, 2011) can deal with the general case of an arbitrary set of hypotheses together with finite action sets. Their definition of regret is different from ours since they compare to the best fixed-parameter solution, whereas we compare to the best action with respect to the changing context. For a general discussion of the advantages of approaches directly taking advantage of structure in contextual bandit problems over the EXP4 family we refer to (Chu et al., 2011). Epoch-Greedy (Langford & Zhang, 2008), which also works in a setting more general than ours, achieves a better dependence on the size of the set of hypotheses but a worse dependence on time T. The VE algorithm (Beygelzimer et al., 2010) which is based on EXP4.P has a regret bound that scales as $O(\sqrt{Td\ln T})$ where d is the VC dimension of the hypothesis class.

Other related work includes (Seldin, Auer, Laviolette, Shawe-Taylor, & Ortner, 2011) which studies a different setting with finite context spaces, showing a regret bound that depends on the mutual information between contexts and actions, and Gaussian process bandits (Grünewälder, Audibert, Opper, & Shawe-Taylor, 2010) and convex bandits (Cesa-Bianchi & Lugosi, 2006) study mostly continuous actions sets.

3 Kernelised UCB

In this section we show how to derive KernelUCB by directly kernelising the LinUCB algorithm. In contrast GP-UCB is motivated from experimental design. The derivation is straightforward and we provide it for convenience and to introduce the notation which is used in the analysis. Our derivation is the combination of the kernel trick (Shawe-Taylor & Cristianini, 2004) and the kernelised version of the Mahalanobis (Haasdonk & Pekalska, 2010).

Kernel methods assume that there exists a mapping $\phi : \mathbb{R}^d \to \mathcal{H}$ that maps the data to a (possibly infinite dimensional) Hilbert space in which a linear relationship can be observed. We call $\mathbb{R}^d$ the *primal space* and $\mathcal{H}$ the associated *reproducing kernel Hilbert space* (RKHS). We use matrix notation to denote the inner product of two elements $h, h' \in \mathcal{H}$, i.e. $h^\intercal h' := \langle h, h' \rangle_{\mathcal{H}}$ and $\|h\| = \sqrt{\langle h, h \rangle_{\mathcal{H}}}$ to denote the RKHS norm. From the mapping ϕ we have the *kernel function*, defined by:

$$k(x, x') := \phi(x)^\intercal \phi(x'),\ \forall x, x' \in \mathbb{R}^d,$$

and the *kernel matrix* of a data set $\{x_1, \ldots, x_t\} \subset \mathbb{R}^d$ given by $K_t := \{k(x_i, x_j)\}_{i,j \le t}$. For our non-linear contextual bandit model we assume the existence of a ϕ for which there exists a $\theta^* \in \mathcal{H}$ such that:

$$\mathbb{E}(r_{a,t} \mid x_{a,t}) = \phi(x_{a,t})^\intercal \theta^*.$$

Taking $a_t^* := \arg\max_{a \in \mathcal{A}} \{\phi(x_{a,t})^\intercal \theta^*\}$ we can define the regret as before in (1). Note that when $\phi \equiv \text{Id}$, we recover the linear bandit case.

To obtain the upper confidence bounds we derive prediction and width estimators for the expected rewards. LinUCB uses estimators built from ridge regression in the primal. Since we assume that our model is linear in the RKHS we show how to build estimators from ridge regression in $\mathcal{H}$. By deriving equivalent dual forms which involve only entries of the kernel matrix we avoid working directly in the possibly infinite dimensional RKHS.

First we take the prediction estimator to be of the form $\hat{\mu}_{a,t+1} = \phi(x_{a,t+1})^\intercal \theta_t$ where θ_t is the minimiser of the regularised least squares loss function:

$$\mathcal{L}(\theta) = \gamma \|\theta\|^2 + \sum_{i=1}^{t-1} (r_i - \phi(x_i)^\intercal \theta)^2 . \qquad (2)$$

We derive a representation of this estimator involving only kernels between context vectors. We denote $\Phi_t = [\phi(x_1)^\intercal, \ldots, \phi(x_{t-1})^\intercal]^\intercal$. Note that the solution of the minimisation problem $\theta_t := \min_{\theta \in \mathcal{H}} \mathcal{L}(\theta)$ satisfies:

$$(\Phi_t^\intercal \Phi_t + \gamma I)\theta_t = \Phi_t^\intercal y_t.$$

Rearranging this equation we obtain:

$$\theta_t = \Phi_t^\intercal \alpha_t \tag{3}$$

where $\alpha_t = \gamma^{-1}(y_t - \Phi_t\theta_t) = \gamma^{-1}(y_t - \Phi_t\Phi_t^\intercal\alpha_t)$, which implies that $\alpha_t = (K_t + \gamma I)^{-1} y_t$. Finally, denoting $k_{x,t} := \Phi_t\phi(x) = [k(x, x_1), \ldots, k(x, x_{t-1})]^\intercal$ we get:

$$\hat{\mu}_{a,t} = k_{x_{a,t},t}^\intercal (K_t + \gamma I)^{-1} y_t. \tag{4}$$

While the computation of θ_t using (3) would require evaluating $\phi(x_i)$ for every data point x_i, the dualised representation of the prediction (4) allows the computation of $\hat{\mu}_{a,t}(x)$ only from objects in the kernel matrix.

Next we construct the widths of the confidence intervals around the prediction. As for linear bandits we find appropriate widths in terms of the Mahalanobis distance of $\phi(x_{a,t})$ from the matrix Φ_t:

$$\hat{\sigma}_{a,t} := \sqrt{\phi(x_{a,t})^\intercal (\Phi_t^\intercal \Phi_t + \gamma I)^{-1} \phi(x_{a,t})}. \tag{5}$$

Once again we motivate this choice of width by noting that it is exactly the variance of the prediction estimator when the noise in the dualised data is standard normal. In order to compute these widths we derive a dualised representation of (5). Our derivation is similar to the kernelisation of the Mahalanobis distance for centered data in (Haasdonk & Pekalska, 2010): Since the matrices $(\Phi_t^\intercal\Phi_t + \gamma I)$ and $(\Phi_t\Phi_t^\intercal + \gamma I)$ are regularised they are strictly positive definite, and therefore:

$$(\Phi_t^\intercal\Phi_t + \gamma I)\Phi_t^\intercal = \Phi_t^\intercal(\Phi_t\Phi_t^\intercal + \gamma I)$$
$$\Phi_t^\intercal(\Phi_t\Phi_t^\intercal + \gamma I)^{-1} = (\Phi_t^\intercal\Phi_t + \gamma I)^{-1}\Phi_t^\intercal.$$

Now we can extract the Mahalanobis distance from the last equation

$$(\Phi_t^\intercal\Phi_t + \gamma I)\phi(x) = (\Phi_t^\intercal k_{x,t} + \gamma\phi(x))$$

from which we deduce that

$$\phi(x) = \Phi_t^\intercal(\Phi_t\Phi_t^\intercal + \gamma I)^{-1}k_{x,t} + \gamma(\Phi_t^\intercal\Phi_t + \gamma I)^{-1}\phi(x)$$

and express $\phi(x)^\intercal\phi(x)$ as

$$k_{x,t}^\intercal(\Phi_t\Phi_t^\intercal + \gamma I)^{-1}k_{x,t} + \gamma\phi(x)^\intercal(\Phi_t^\intercal\Phi_t + \gamma I)^{-1}\phi(x).$$

Rearranging we get an expression for the width involving only inner products:

$$\hat{\sigma}_{a,t} := \gamma^{-1/2}\sqrt{k(x_{a,t}, x_{a,t}) - k_{x_{a,t},t}^\intercal (K_t + \gamma I)^{-1} k_{x_{a,t},t}}.$$

Algorithm 1 KernelUCB with online updates

```
Input and initialisation:
    $N$ the number of actions, $T$ the number of pulls,
    $\gamma$, $\eta$ regularization and exploration parameters
    $k(\cdot,\cdot)$ kernel function
$u_0 \leftarrow [1, 0, ..., 0]^\intercal$ (at start first action is pulled)
$y_0 \leftarrow \emptyset$
Run:
for $t = 1$ to $T$ do
    Choose $a \leftarrow \arg\max u_{t-1}$ and get reward $r_{t-1}$
    Update $y_t \leftarrow [r_1, \ldots, r_{t-1}]^\intercal$
  if $t = 1$ then
      $K_t^{-1} \leftarrow 1/k_{x_t,x_t} + \gamma$
  else {online update of the kernel matrix inverse}
      $b \leftarrow (k_{x_1}, k_{x_2}, \ldots, k_{x_{t-1}})^\intercal$
      $K_{22} \leftarrow (k_{x_a,x_a} + \gamma - b^\intercal K_{t-1}^{-1} b)^{-1}$
      $K_{11} \leftarrow K_{t-1}^{-1} + K_{22}K_{t-1}^{-1}bb^\intercal K_{t-1}^{-1}$
      $K_{12} \leftarrow -K_{22}K_{t-1}b$
      $K_{21} \leftarrow -K_{22}b^\intercal K_{t-1}^{-1}$
      $K_t^{-1} \leftarrow [K_{11}, K_{12}; K_{21}, K_{22}]$
  end if
  for $a = 1$ to $N$ do
      $\sigma_{a,t} \leftarrow \sqrt{k(x_{a,t}, x_{a,t}) - k_{x,t}^\intercal K_t^{-1} k_{x,t}}$
      $u_{a,t} \leftarrow \left(k_{x,t}^\intercal K_t^{-1} y_t + \frac{\eta}{\gamma^{1/2}}\sigma_{a,t}\right)$
  end for
end for
```

As for LinUCB, KernelUCB chooses the action a_t at time t which satisfies

$$a_t := \arg\max_{a \in A}(k_{x_{a,t},t}^\intercal (K_t + \gamma I_t)^{-1} y_t + \\ + \frac{\eta}{\gamma^{1/2}}\sqrt{k(x_{a,t}, x_{a,t}) - k_{x_{a,t},t}^\intercal (K_t + \gamma I)^{-1} k_{x_{a,t},t}}),$$

where η is a (possibly time dependent) exploration parameter of the algorithm. Considering a_t and $\hat{\sigma}_{a,t}$ we see that GP-UCB is a special case of KernelUCB where the regularization constant is set to the model noise.

The selection of an appropriate kernel function is problem dependent (Shawe-Taylor & Cristianini, 2004). The linear kernel corresponds to $\phi \equiv \text{Id}$ and leads to the dual representation of the LinUCB algorithm in the primal. A non-linear kernel function creates a kernelised UCB algorithm for a non-linear bandit. Typical examples of non-linear kernel functions include: the radial basis function where $k(x_i, x_j) = \exp(-||x_i - x_j||^2/2\sigma^2)$, for $\sigma > 0$ and the polynomial kernel $k(x_i, x_j) = (x_i^\intercal x_j + 1)^p$. The pseudocode of KernelUCB is displayed in Algorithm 1 and uses the inversion update of K_t through the properties of the Schur complement (Zhang, 2005).

Algorithm 2 SupKernelUCB

```
Input and initialisation:
  $T$ number of arm pulls, $S$ number of sets
$\Psi_1^{(s)} \leftarrow \emptyset$ for all $s \in [T]$
for $t = 1$ to $T$ do
  $s \leftarrow 1$ and $\hat{A}_1 \leftarrow [N]$
  repeat
    $\left(\hat{\mu}_{t,a}^{(s)}, \hat{\sigma}_{t,a}^{(s)}\right) \leftarrow$ BaseKernelUCB with $\Psi_t^{(s)}$ for all $a \in \hat{A}_{(s)}$
    if $\eta\hat{\sigma}_{t,a}^{(s)} \leq 1/\sqrt{T}$ for all $a \in \hat{A}_{(s)}$ then
      Choose $a_t = \arg\max_{a \in \hat{A}_{(s)}} \left(\hat{\mu}_{t,a}^{(s)} + \eta\hat{\sigma}_{t,a}^{(s)}\right)$
      Keep the sets $\Psi_{t+1}^{(s')} = \Psi_t^{(s')}$ for all $s' \in [S]$
    else
      if $\eta\hat{\sigma}_{t,a}^{(s)} \leq 2^{-s}$ for all $a \in \hat{A}_{(s)}$ then
        $\hat{A}_{(s+1)} \leftarrow \{a \in \hat{A}_{(s)} | \hat{\mu}_{t,a}^{(s)} + \eta\hat{\sigma}_{t,a}^{(s)} \geq \max_{a' \in \hat{A}_{(s)}} \hat{\mu}_{t,a'}^{(s)} + \eta\hat{\sigma}_{t,a'}^{(s)} - 2^{1-s}\}$
        $s \leftarrow s + 1$
      else
        Choose $a_t \in \hat{A}_{(s)}$ such that $\eta\hat{\sigma}_{t,a_t}^{(s)} > 2^{-s}$.
        Update the index sets at all levels $\Psi_{t+1}^{(s')} = \begin{cases} \Psi_t^{(s')} \cup \{t\} & \text{if } s = s \\ \Psi_t^{(s')} & \text{otherwise.} \end{cases}$
      end if
    end if
  until an action $a_t$ is found
end for
```

4 Analysis

In this section we provide an upper bound on the cumulative regret defined in Section 2 for KernelUCB. As for LinUCB, the predictors for KernelUCB, $\hat{\mu}_{a,t}$, are sums of *dependent* random variables. Consequently, we are unable to directly apply the Azuma-Hoeffding inequality to gain control over the error in the predictors. To get around this problem we use the construction of Auer (2002) and introduce the related algorithm SupKernelUCB, the appropriate modification of KernelUCB. SupKernelUCB (Algorithm 2) constructs special, mutually exclusive subsets $\{\Psi_t^{(s)}\}_s$ of the elapsed time. On each of these sets it builds predictors, $\hat{\mu}_{a,t}^{(s)}$, and widths, $\hat{\sigma}_{a,t}^{(s)}$, in the same way that KernelUCB does, using the BaseKernelUCB (Algorithm 3) subroutine. In the pseudocodes and below, $[n]$ denotes the set $\{1, \ldots, n\}$. At the beginning of the algorithm all the subsets $\{\Psi_1^{(s)}\}_s$ are initialised to the empty set, and at each time $t \geq 1$ the value t is included in at most one $\{\Psi_{t+1}^{(s)}\}_s$ in such a way that the event $\{t \in \Psi_{t+1}^{(s)}\}$ is independent of the rewards observed at times in $\Psi_t^{(s)}$. In this way the Azuma-

Algorithm 3 BaseKernelUCB

```
Input and initialisation:
  $\Psi_t \subseteq \{1, 2, \ldots, t-1\}$
  $k(\cdot,\cdot)$ kernel function, $\gamma$ regularization parameter
$K \leftarrow [k(x_i, x_j)]_{i,j \in \Psi_t} + \gamma I$
$y \leftarrow [r_\tau]_{\tau \in \Psi_t}$
for $a \in [N]$ do
  $\hat{\mu}_{t,a} \leftarrow k_{x_{a_t},t}^\intercal K^{-1} y_t$
  $\hat{\sigma}_{t,a} \leftarrow \gamma^{-1/2}\sqrt{k(x_{a_t}, x_{a_t}) - x_{a_t}^\intercal K^{-1} x_{a_t}}$
end for
```

Hoeffding inequality can be applied on each subset $\Psi_t^{(s)}$ to get a regret bound.

If we directly applied known regret bounds (Auer, 2002; Chu et al., 2011) for linear contextual bandits to our setting, we would obtain a bound in terms of the dimension of the RKHS, which is possibly infinite. We avoid this problem through a careful consideration of the eigenvalues of the covariance matrix and the choice of the regularisation constant and give a bound in terms of a data dependent quantity $\tilde{d}$ which we call the *effective dimension*: Let $(\lambda_{i,t})_{i \geq 1}$ denote the eigenvalues of $C_t^\gamma = \Phi_t^\intercal \Phi_t + \gamma I$ in decreasing order and define:

$$\tilde{d} := \min\{j : j\gamma \ln T \geq \Lambda_{T,j}\} \text{ where } \Lambda_{T,j} := \sum_{i>j} \lambda_{i,T} - \gamma.$$

Theorem 1 *Assume that $\|\phi(x_{a,t})\| \leq 1$ and $|r_{a,t}| \in [0,1]$ for all $a \in A$ and $t \geq 1$, and set $\eta = \sqrt{2\ln 2TN/\delta}$. Then with probability $1-\delta$, SupKernelUCB satisfies:*

$$R(T) \leq \left[2 + 2\left(1 + \sqrt{\frac{\gamma}{2\ln(2TN(1+\ln T)/\delta)}}\right)\|\theta^*\| + \right.$$
$$+ 8\sqrt{\left(12 + \frac{15}{\gamma}\right)\max\left\{\ln\left(\frac{T}{\tilde{d}\gamma} + 1\right), \ln T\right\}^3} \times$$
$$\left. \times \sqrt{\left(2\ln\frac{2TN(1+\ln T)}{\delta}\right)}\right]\sqrt{\tilde{d}T}$$

Remark 1 *We call $\tilde{d}$ the effective dimension because it gives a proxy for the number of principle directions over which the projection of the data in the RKHS is spread. If the data all fall within a subspace of $\mathcal{H}$ of dimension d', then $\Lambda_{T,d'} = 0$ and $\tilde{d} \leq d'$. However more generally $\tilde{d}$ can be thought of as a measure of how quickly the eigenvalues of $\Phi_t^\intercal \Phi_t$ are decreasing. For example if the eigenvalues are only polynomially decreasing in i (i.e. $\lambda_i \leq Ci^{-\alpha}$ for some $\alpha > 1$ and some constant $C > 0$) then $\tilde{d} \leq 1 + (C/(\gamma \ln T))^{1/\alpha}$.*

Remark 2 *When $\Phi \equiv \mathrm{Id}$, $\tilde{d} \leq d$, the assumption that $\|\phi(x_{a,t})\| \leq 1$ becomes the assumption that the contexts* *are normalised in the primal, and we recover exactly the result from (Chu et al., 2011) which matches the lower bound for this setting.*

Remark 3 *Theorem 1 suggests that if we know that $\|\theta^*\| \leq L$, for some L, we should set γ to be of the order of L^{-1} so that we obtain $\tilde{O}(\sqrt{L\tilde{d}T})$ regret. If we do not have such knowledge, just setting γ to a constant (e.g., found by a cross-validation) will incur $\tilde{O}(\|\theta^*\|\sqrt{\tilde{d}T})$ regret.*

The proof of this theorem follows the scheme of the proof of Theorem 1 in (Chu et al., 2011). The first step is to prove a high probability bound on the error in the predictors $\hat{\mu}^{(s)}_{a,t}$, and to do this we use a classical concentration result, the Azuma-Hoeffding inequality. Our result here generalises Lemma 1 of Chu et al. (2011) to 1) linear products in RKHS, 2) regularisation γ, and 3) no assumption that $\|\theta^*\| \leq 1$. Also the trade-off between γ and $\|\theta^*\|$ becomes evident. For ease of notation, in the below we drop the superscript (s) whenever it is superfluous.

Lemma 1 *Suppose that the conditions of Theorem 1 hold, and that the input index set $\Psi^{(s)}_t$ for BaseKernelUCB is constructed so that for fixed contexts $x_{a_\tau,\tau}$, $\tau \in \Psi^{(s)}_t$, the rewards $r_{a_\tau,\tau}$ are independent random variables. Then with probability at least $1 - 2Ne^{-\eta^2/2}$ we have for all $a \in A$:*

$$|\hat{\mu}^{(s)}_{a,t} - \phi(x_{a,t})^\intercal\theta^*| \leq (\eta(1 + \|\theta^*\|) + \gamma^{1/2}\|\theta^*\|)\hat{\sigma}^{(s)}_{a,t}.$$

Proof. We begin by noting that:

$$\begin{aligned}&\hat{\mu}_{a,t} - \phi(x_{a,t})^\intercal\theta^* \qquad (6)\\ &= \phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}\Phi^\intercal_t y_t - \phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}(\Phi^\intercal_t\Phi_t + \gamma I)\theta^*\\ &= \phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}\Phi^\intercal_t(y_t - \Phi_t\theta*) - \gamma\phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}\theta^*\end{aligned}$$

Now by construction of the set Ψ_t we know that $(y_t - \Phi_t\theta^*) \mid \Phi_t, x_{a,t}$ is a vector of zero mean independent random variables. Hence we can apply the Azuma-Hoeffding inequality to obtain that:

$$\begin{aligned}&\mathbb{P}(|\phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}\Phi^\intercal_t(y_t - \Phi_t\theta^*)| \qquad (7)\\ &\qquad > (1 + \|\theta^*\|)\eta\hat{\sigma}_{a,t}) \leq 2e^{-\eta^2/2},\end{aligned}$$

since $|r_\tau - \phi(x_\tau)^\intercal\theta^*| \leq 1 + \|\theta^*\|$ for any τ and

$$\begin{aligned}\hat{\sigma}^2_{a,t} &= \phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}\phi(x_{a,t})\\ &= \phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}(\Phi^\intercal_t\Phi_t + \gamma I)(C^\gamma_t)^{-1}\phi(x_{a,t})\\ &\geq \|\Phi_t(C^\gamma_t)^{-1}\phi(x_{a,t})\|^2.\end{aligned}$$

Now by the Cauchy-Schwarz inequality we find that:

$$\begin{aligned}&|\phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}\theta^*| \leq\\ &\leq \|\theta^*\|\sqrt{\phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}\gamma^{-1}\gamma I(C^\gamma_t)^{-1}\phi(x_{a,t})}\\ &\leq \gamma^{-1/2}\|\theta^*\|\sqrt{\phi(x_{a,t})^\intercal(C^\gamma_t)^{-1}C^\gamma_t(C^\gamma_t)^{-1}\phi(x_{a,t})}\\ &\leq \gamma^{-1/2}\|\theta^*\|\hat{\sigma}_{a,t}. \qquad (8)\end{aligned}$$

The result follows by plugging (7) and (8) into (6). ■

The second step of the analysis bounds the widths $\hat{\sigma}^{(s)}_{a,t}$ in terms of the change in eigenvalues of the matrix C^γ_t. To do this we extend the argument in Lemma 11 by Auer (2002) to possibly infinite matrices. Let us define $\psi_{s,t} := |\Psi^{(s)}_t|$.

Lemma 2 *The eigenvalues of $\Phi^\intercal_t\Phi_t$ do not depend on the choice of basis for $\mathcal{H}$. Moreover the representation of $\Phi^\intercal_t\Phi_t$ in any basis $\mathcal{B}$ created by extending a maximal linearly independent subset of $\{\phi(x_{a,\tau})\}_{\tau\in\Psi_t}$ has zeros everywhere outside its top-left $(\psi_t \times \psi_t)$-submatrix.*

Proof. Assume that $\phi = \phi_\mathcal{E}$ is described in terms of some basis $\mathcal{E}$ for $\mathcal{H}$. Let $\mathcal{B}$ be any basis for $\mathcal{H}$ extended from a maximal linearly independent subset of $\{\phi(x_{a,s})\}_{s\leq t}$. If $Q_{\mathcal{BE}}$ denotes the change of basis matrix from $\mathcal{B}$ to $\mathcal{E}$ then $\Phi_{\mathcal{E},t} = \Phi_{\mathcal{B},t}Q_{\mathcal{BE}}$ and:

$$\Phi^\intercal_{\mathcal{E},t}\Phi_{\mathcal{E},t} = Q^\intercal_{\mathcal{BE}}\Phi^\intercal_{\mathcal{B},t}\Phi_{\mathcal{B},t}Q_{\mathcal{BE}},$$

where $\Phi_{\mathcal{B},t}$ and $\Phi_{\mathcal{E},t}$ denote the matrix Φ_t with respect to the bases $\mathcal{B}$ and $\mathcal{E}$. Moreover the (i,j)-th entry of $\Phi^\intercal_{\mathcal{B},t}\Phi_{\mathcal{B},t}$ is zero when $\max\{i,j\} > \psi_t$. Hence the eigenvalues are independent of the choice of basis and only the first t of them can be non-zero. ■

Lemma 3 *Suppose that $\Psi^{(s)}_{t+1} = \Psi^{(s)}_t \cup \{t\}$. Then the eigenvalues of C^γ_t can be arranged so that $\lambda_{j,t-1} \leq \lambda_{j,t}$ for each $j \geq 1$, and:*

$$\hat{\sigma}^{(s)}_{a,t} \leq \sqrt{\left(4 + \frac{6}{\gamma}\right)\sum_{j=1}^{\psi_{s,t+1}}\frac{\lambda_{j,t} - \lambda_{j,t-1}}{\lambda_{j,t-1}}}.$$

where $\lambda_{j,0} := \gamma$ for all j.

Proof. Let $\mathcal{B}$ be a basis defined as in Lemma 2, let $C_{\mathcal{B},t} := \Phi^\intercal_{\mathcal{B},t}\Phi_{\mathcal{B},t}$, and let $\tilde{C}_t$ denote the top-left, $\psi_{s,t+1} \times \psi_{s,t+1}$-submatrix of $C_{\mathcal{B},t}$ and let $\tilde{\phi}(x_t)$ denote the first t entries in the vector $\phi_\mathcal{B}(x_t)$. It follows from Lemma 2:

$$C_{\mathcal{B},t+1} + \gamma I = \left(\begin{array}{c|c}\tilde{C}_t & 0\\ \hline 0 & 0\end{array}\right) + \left(\begin{array}{c|c}\tilde{\phi}(x_t)\tilde{\phi}(x_t)^\intercal & 0\\ \hline 0 & 0\end{array}\right) + \gamma I$$

and we may apply the argument of the proof of Lemma 11 by Auer (2002) to the top left $\psi_{s,t+1} \times \psi_{s,t+1}$ blocks to obtain the result. Note that we only need to sum up to $\psi_{s,t+1}$ because $\lambda_j = \gamma$ for all $j > \psi_{s,t+1}$. ■

The third step of the analysis uses the bound on the widths in Lemma 3 to bound their sum. Since our matrices C_t^γ are possibly infinite we use the effective dimension of the data, $\tilde{d}$, to reduce the analysis to the finite dimensional case.

Lemma 4 *Let* $l_T = \max\{\ln(T/(\tilde{d}\gamma)+1), \ln T\}$. *Then:*

$$\sum_{t \in \Psi_{T+1}^{(s)}} \hat{\sigma}_{a,t}^{(s)} \leq \sqrt{\left(10 + \frac{15}{\gamma}\right) \tilde{d} T l_T} \text{ for all } s \in [S].$$

Proof. From Lemma 3 we know that the eigenvalues of C_t^γ can be arranged so that $\lambda_{j,t-1} \leq \lambda_{j,t}$ for each $j \geq 1$. Once such an arrangement exists we can always rearrange the eigenvalues so that they are also decreasing in j, for each t. By Lemma 3 we have:

$$\begin{aligned}
\sum_{t \in \Psi_{T+1}} \hat{\sigma}_{a,t} &\leq \sum_{t=1}^{\psi_{T+1}} \sqrt{\sum_{j=1}^{\psi_{T+1}} \frac{\lambda_{j,t} - \lambda_{j,t-1}^{(s)}}{\lambda_{j,t-1}}} \\
&\leq \sum_{t=1}^{\psi_{T+1}} \left[\sqrt{\sum_{1 \leq j \leq \tilde{d}} \frac{\lambda_{j,t} - \lambda_{j,t-1}}{\lambda_{j,t-1}}} + \sqrt{\sum_{j=\tilde{d}+1}^{\psi_{T+1}} \frac{\lambda_{j,t} - \lambda_{j,t-1}}{\lambda_{j,t-1}}} \right] \\
&\leq \underbrace{\sum_{t=1}^{\psi_{T+1}} \sqrt{\sum_{1 \leq j \leq \tilde{d}} \frac{\lambda_{j,t} - \lambda_{j,t-1}}{\lambda_{j,t-1}}}}_{(A)} + \underbrace{\sqrt{\frac{\psi_{T+1}}{\gamma} \sum_{j \geq \tilde{d}+1} \lambda_{j,T}}}_{(B)},
\end{aligned}$$

where we have used the Cauchy-Schwarz inequality for the second inequality. Now by the definition of $\tilde{d}$, term (B) is bounded by $\sqrt{\tilde{d}\psi_{T+1} \ln T}$. Writing $\alpha_{i,t} = \lambda_{i,t} - \lambda_{i,t-1}$, term (A) becomes:

$$\sum_{t=1}^{\psi_{T+1}} \sqrt{\sum_{i=1}^{\tilde{d}} \frac{\alpha_{i,t}}{\sum_{s=1}^{t} \alpha_{i,s} + \gamma}}$$

where $\lambda_{i,0} = \gamma$ and $\sum_{i=1}^{\tilde{d}} \alpha_{i,t} \leq tr(C_t^\gamma) - tr(C_{t-1}^\gamma) = \|\phi(x_{a,t})\|^2 \leq 1$. We upper bound this object by solving an easier maximisation problem:

$$\max_{(\alpha)_{i,t}, (\varepsilon)_{i,t}} \sum_{t=1}^{\psi_{T+1}} \sqrt{\sum_{i=1}^{\tilde{d}} \frac{\alpha_{i,t}}{\sum_{s=1}^{t} \varepsilon_{i,s} + \gamma}}$$

under the constraints $\sum_i \alpha_{i,t} = \sum_i \epsilon_{i,t} \leq 1$. Using the method of Lagrange multipliers and the Cauchy-Schwarz inequality[1] we can upper bound the solution to this maximisation problem by $\sqrt{\tilde{d}\psi_{T+1} \log(\psi_{T+1}/(\tilde{d}\gamma) + 1)}$. We conclude by noting that $\psi_{T+1} \leq T$. ■

[1]We do not include a detailed derivation due to the space constraints.

The fourth step is to bound the size of the sets $\Psi_{T+1}^{(s)}$. This is achieved by plugging our Lemma 4 into the proof of Lemma 16 in (Auer, 2002):

Lemma 5 *For all* $s \in [S]$*:*

$$\psi_{s,T+1} \leq 2^s \eta \sqrt{\left(10 + \frac{15}{\gamma}\right) \tilde{d} \psi_{s,T+1} l_T}$$

The lemmas above have analysed the properties of BaseKernelUCB assuming independence. The effect of the SupKernelUCB construction is described in Lemma 14 and 15 by Auer (2002), which we restate for convenience using our notation. Lemma 6 shows that there the trials given to BaseKernelUCB are indeed independent:

Lemma 6 *For each* $s \in [S]$, *each* $t \in [T]$, *and any fixed sequence of feature vectors* x_t *with* $t \in \Psi_t^{(s)}$, *the corresponding rewards* r_t *are independent random variables such that* $\mathbb{E}(r_t) = \phi(x_t)^\intercal \theta^*$.

Lemma 7 gives the properties of SupKernelUCB needed to provide the final regret bound, where the first item is a consequence of Lemma 1:

Lemma 7 *With probability* $1 - 2Ne^{-\eta^2/2}$, *for any* $t \in [T]$ *and any* $s \in [S]$, *the following hold:*

- $|\hat{\mu}_{a,t}^{(s)} - \phi(x_{a,t})^\intercal \theta^*| \leq (\eta(1 + \|\theta^*\|) + \gamma^{1/2}\|\theta^*\|)\hat{\sigma}_{a,t}^{(s)}$, *for all* $a \in [N]$
- $a_t^* \in \hat{A}_s$, *and*
- $\mathbb{E}[r_{a_t^*,t}] - \mathbb{E}[r_t] \leq 2^{3-s}$ *for any* $a \in \hat{A}_s$.

Now we find the final regret bound with a similar scheme as Auer (2002) using all the previous lemmas.

Proof of Theorem 1. First we upper bound the expected regret $\mathbb{E}[R(T)]$ with probability at least $1 - 2NTe^{-\eta^2/2}$ with:

$$\begin{aligned}
\sum_{t \in [T] \setminus \Psi_0} \mathbb{E}(r_{a_t^*,t}) - \mathbb{E}(r_t) &= \sum_{s=1}^{S} \sum_{t \in \Psi_{T+1}^{(s)}} \mathbb{E}(r_{a_t^*,t}) - \mathbb{E}(r_t) \\
&\leq \sum_{s=1}^{S} 2^{3-s} \psi_{s,T+1} \leq \eta S \sqrt{\left(10 + \frac{15}{\gamma}\right) \tilde{d} \psi_{s,T+1} l_T} \quad (9)
\end{aligned}$$

and:

$$\sum_{t \in \Psi_0} \mathbb{E}(r_{a_t^*,t}) - \mathbb{E}(r_t) \leq 2\left(2 + \|\theta^*\| + \frac{\gamma^{\frac{1}{2}}\|\theta^*\|}{\eta}\right)\sqrt{T}, \quad (10)$$

where $\Psi_0 := [T] \setminus \bigcup_{s\in[S]} \Psi_{T+1}^{(s)}$. In (9) we have used Lemma 7 for the first inequality and Lemmas 5 and 6 for the second inequality. In (10) we used Lemma 1 and the construction of the Ψ_0 set in SupKernelUCB. Now a standard application of the Azuma-Hoeffding inequality tells us that, with probability at least $1 - 2TNe^{-\eta^2/2}$:

$$|R(T) - \mathbb{E}(R(T|H_{T-1}))| \leq \sqrt{2T\ln(1/TNe^{-\eta^2/2})}. \quad (11)$$

Finally setting $S = \ln T$, $\eta = \sqrt{2\ln 2TN/\delta}$, and bounding $\psi_{s,T+1}$ by T we obtain from (9), (10), and (11):

$$\begin{aligned} R(T) &= \mathbb{E}(R(T))) + [R(T) - \mathbb{E}(R(T|H_{T-1}))] \\ &\leq 8\sqrt{\left(10 + \frac{15}{\gamma}\right) l_T^3 \left(2\ln\frac{2TN}{\delta}\right)}\sqrt{\tilde{d}T} \\ &\quad + 2\left(1 + \left(1 + \sqrt{\frac{\gamma}{2\ln(2TN/\delta)}}\right)\|\theta^*\|\right)\sqrt{T} \\ &\quad + \sqrt{2\ln(1/\delta)}\sqrt{T}. \end{aligned}$$

with probability $1 - (1 + \ln T)\delta$. The result follows by substituting $\delta/(1 + \ln T)$ for δ. ■

4.1 Relationship with GP-UCB

We now relate our analysis to that of GP-UCB in (Srinivas et al., 2010), and in particular to their Theorem 3, which treats the agnostic case. In this case, θ^* is not assumed to be sampled from a GP, but instead to have a bounded RKHS norm $\|\theta^*\|$. Under this assumption, the cumulative regret is bounded as:

$$O\Big(\big(I(y_A;\theta^*) + \|\theta^*\|^2\sqrt{I(y_A;\theta^*)}\big)\sqrt{T}\Big), \quad (12)$$

where $I(y_T;\theta^*)$ is the mutual information between θ^* and the vector of (noisy) observations y_T. Both $I(y_T;\theta^*)$ in (12) and $\tilde{d}$ are data dependent quantities. We now relate them in order to compare the analyses.

We have that:

$$\begin{aligned} I(y_T;f) &= \ln|I + \sigma^{-2}K_T| = \sum_i \ln(1 + \sigma^{-2}\lambda_{i,T}) \\ &\geq \ln\left(1 + \sigma^{-2}\lambda_{\tilde{d}-1,T}\right)\left(\tilde{d} - 1 + \frac{\sum_{i>\tilde{d}-1}\lambda_{i,T}}{\lambda_{\tilde{d}-1,T}}\right) \\ &\geq (\tilde{d} - 1)\ln\left(1 + \sigma^{-2}\lambda_{\tilde{d}-1,T}\right)\left[1 + \frac{\gamma\ln T}{\lambda_{\tilde{d}-1,T}}\right] \\ &\geq (\tilde{d} - 1)\max_B \min\Big\{\ln(1 + B)\gamma\sigma^{-2}\ln(T), \frac{\ln(1 + B)}{B}\Big\} \\ &\geq \Omega(\tilde{d}\ln\ln T) \end{aligned}$$

In the second equality, we used the fact that the eigenvalues of $\Phi_T^\intercal\Phi_T$ are the same as the eigenvalues of $\Phi_T\Phi_T^\intercal$. In the second inequality we used the definition of $\tilde{d}$. For the last inequality we considered the two cases when $\lambda_{\tilde{d}-1,T} \leq B\sigma^2$ and when $\lambda_{\tilde{d}-1,T} \geq B\sigma^2$ for some B.

This shows that $\tilde{d}$ is at least as good as $I(y_T;\theta^*)$, and comparing our Theorem 1 with (12), our regret bound only scales as $O(\sqrt{\tilde{d}})$, while the dependence of the regret bound (12) is linear in $I(y_T;\theta^*)$. In particular, this means that for the linear kernel we attain the lower bound for linear contextual bandits, (Chu et al., 2011), while GP-UCB is $\sqrt{d}$ away. This concerns only the agnostic case of GP-UCB, i.e. Theorem 3 in (Srinivas et al., 2010), which is the same setting as ours. When θ^* is sampled from a GP, their result for linear case also matches the lower bound.

Srinivas et al. (2010) also provide an upper bound on $I(y_T;\theta^*)$, denoted by γ_T, for certain kernels. As a consequence of the link between $I(y_T;\theta^*)$, γ_T and $\tilde{d}$, we may also express our bounds in terms of γ_T. Moreover, in the agnostic case again, our bounds enjoy an improved dependence on this parameter: for example, for the widely used RBF kernel, our bound scales with $O(\ln T)^{d/2}$ in place of $O(\ln T)^d$.

Finally, when $\|\theta^*\|$ is unknown and we are unable to regularise appropriately, our regret bound only depends on $\|\theta^*\|$ linearly (Remark 3), while the dependence in (12) is quadratic.

5 Conclusion

We derive and analyse KernelUCB, an algorithm for contextual bandits, which is able to run with just a similarity function instead of context features. We give a finite-time theoretical analysis that proves the cumulative regret scales as $\tilde{O}(\sqrt{T\tilde{d}})$ where $\tilde{d}$ is the effective dimension of the data in the feature space.

As a special case of our algorithm and its analysis, we recover the known upper bound for LinUCB that matches the lower bound for the linear problem. In the case when we know an upper bound on the model noise, then setting our regulariser to that value recovers the GP-UCB algorithm. Moreover, we provide an improved analysis for the agnostic case, when the reward function is not necessarily sampled from a GP prior. Finally, our analysis shows the dependence of the regulariser on the RKHS norm of reward function.

6 Acknowledgements

This research was funded by European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 270327 (project CompLACS).

References

Abbasi-Yadkori, Y., Pal, D., & Szepesvari, C. (2011). Improved Algorithms for Linear Stochastic Bandits. In *Advances in Neural Information Processing Systems 24*, pp. 2312–2320.

Auer, P. (2002). Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, *3*, 397–422.

Auer, P., Cesa-Bianchi, N., Freund, Y., & Schapire, R. E. (2003). The Nonstochastic Multiarmed Bandit Problem. *SIAM Journal on Computing*, *32*(1), 48–77.

Beygelzimer, A., Langford, J., Li, L., Reyzin, L., & Schapire, R. E. (2010). Contextual Bandit Algorithms with Supervised Learning Guarantees. *Machine Learning*, *15*, 14.

Bubeck, S., & Cesa-Bianchi, N. (2012). Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning*, *5*, 1–122.

Cesa-Bianchi, N., & Lugosi, G. (2006). *Prediction, Learning, and Games.* Cambridge University Press, New York, NY.

Chen, Y., Garcia, E. K., Gupta, M. R., Rahimi, A., & Cazzanti, L. (2009). Similarity-based Classification: Concepts and Algorithms. *Journal of Machine Learning Research*, *10*(206), 747–776.

Chu, L., Li, L., Reyzin, L., & Schapire, R. E. (2011). Contextual Bandits with Linear Payoff Functions. In *Proceedings of the 14th International Conference on Articial Intelligence and Statistics.*

Dani, V., Hayes, T. P., & Kakade, S. M. (2008). Stochastic Linear Optimization under Bandit Feedback. In *The 21st Annual Conference on Learning Theory*, pp. 355–366.

Dudik, M., Hsu, D., Kale, S., Karampatziakis, N., Langford, J., Reyzin, L., & Zhang, T. (2011). Efficient Optimal Learning for Contextual Bandits. *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence.*

Grünewälder, S., Audibert, J.-Y., Opper, M., & Shawe-Taylor, J. (2010). Regret Bounds for Gaussian Process Bandit Problems. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics.*

Haasdonk, B., & Pekalska, E. (2010). Classification with Kernel Mahalanobis Distance Classifiers. *Advances in Data Analysis, Data Handling and Business Intelligence*, 351–361.

Kleinberg, R., Slivkins, A., & Upfal, E. (2008). Multi-armed bandit problems in metric spaces. In *Proceedings of the 40th ACM symposium on Theory Of Computing*, pp. 681–690.

Krause, A., & Ong, C. S. (2011). Contextual Gaussian Process Bandit Optimization. In *Proceedings of Neural Information Processing Systems (NIPS).*

Lai, T. L., & Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, *6*(1), 4–22.

Langford, J., & Zhang, T. (2008). The Epoch-Greedy Algorithm for Multi-armed Bandits with Side Information. In Platt, J. C., Koller, D., Singer, Y., & Roweis, S. (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 817–824. MIT Press, Cambridge, MA.

Li, L., Chu, W., Langford, J., & Schapire, R. E. (2010). A Contextual-Bandit Approach to Personalized News Article Recommendation. *WWW 10*, *173*, 10.

Lu, T., Pál, D., & Pál, M. (2010). Contextual Multi-Armed Bandits. In Teh, Y. W., & Titterington, M. (Eds.), *Proceedings of the 13th international conference on Artificial Intelligence and Statistics*, Vol. 9, pp. 485–492.

Rusmevichientong, P., & Tsitsiklis, J. N. (2010). Linearly Parameterized Bandits. *Math. Oper. Res.*, *35*(2), 395–411.

Seldin, Y., Auer, P., Laviolette, F., Shawe-Taylor, J. S., & Ortner, R. (2011). PAC-Bayesian Analysis of Contextual Bandits. In *Neural Information Processing Systems (NIPS)*, pp. 1683–1691.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel Methods for Pattern Analysis.* Cambridge University Press.

Slivkins, A. (2009). Contextual Bandits with Similarity Information. *Proceedings of the 24th annual Conference On Learning Theory*, 1–27.

Srinivas, N., Krause, A., Kakade, S., & Seeger, M. (2010). Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. *Proceedings of International Conference on Machine Learning*, 1015–1022.

Steinberger, R., Pouliquen, B., & Van der Goot, E. (2009). An Introduction to the {Europe Media Monitor} Family of Applications. In *Information Access in a Multilingual World-Proceedings of the SIGIR 2009 Workshop (SIGIR-CLIR'2009)*, pp. 1–8.

Zhang, F. (2005). *The Schur complement and its applications*, Vol. 4. Springer.

Dynamic Blocking and Collapsing for Gibbs Sampling

Deepak Venugopal
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, USA
dxv021000@utdallas.edu

Vibhav Gogate
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, USA
vgogate@hlt.utdallas.edu

Abstract

In this paper, we investigate combining blocking and collapsing – two widely used strategies for improving the accuracy of Gibbs sampling – in the context of probabilistic graphical models (PGMs). We show that combining them is not straight-forward because collapsing (or eliminating variables) introduces new dependencies in the PGM and in computation-limited settings, this may adversely affect blocking. We therefore propose a principled approach for tackling this problem. Specifically, we develop two scoring functions, one each for blocking and collapsing, and formulate the problem of partitioning the variables in the PGM into blocked and collapsed subsets as simultaneously maximizing both scoring functions (i.e., a multi-objective optimization problem). We propose a dynamic, greedy algorithm for approximately solving this intractable optimization problem. Our dynamic algorithm periodically updates the partitioning into blocked and collapsed variables by leveraging correlation statistics gathered from the generated samples and enables rapid mixing by blocking together and collapsing highly correlated variables. We demonstrate experimentally the clear benefit of our dynamic approach: as more samples are drawn, our dynamic approach significantly outperforms static graph-based approaches by an order of magnitude in terms of accuracy.

1 Introduction

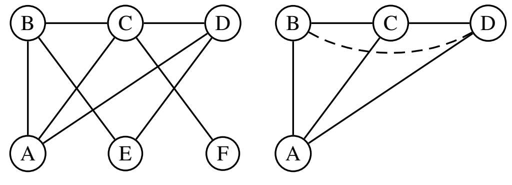
Blocking [1, 2] and collapsing [2] are two popular strategies for improving the statistical efficiency of Gibbs sampling [3] – arguably the most widely used approximate inference scheme for probabilistic graphical models (PGMs). Both these strategies trade sample quality with sample size. The hope is that the user will achieve the right balance between the two for the specific PGM at hand, improving the estimation accuracy as a result.

Unlike Gibbs sampling which samples each variable individually given others, blocked Gibbs sampling partitions the variables into disjoint groups or blocks and then *jointly samples* all variables in each block given an assignment to all other variables not in the block. Joint sampling is more expensive than sampling variables individually but the samples are of higher quality in that for a fixed sample size, the estimates based on blocked Gibbs sampling have smaller variance than the ones based on Gibbs sampling [2]. A collapsed Gibbs sampler[1] operates by *marginalizing out* a subset of variables (collapsed variables) and then generating dependent samples from the marginal distribution over the remaining variables via conventional Gibbs sampling. Marginalizing out variables is more expensive than sampling them. However, since only a sub-space is sampled, the samples are of higher quality.

Although, it is provably better to collapse a variable rather than block (group) it with other variables [2], collapsing is computationally more expensive than blocking and in practice, in many cases, the latter is feasible while the former is not. Therefore, an obvious idea is to combine blocking and collapsing, and the purpose of this paper is to investigate this combination in the context of PGMs. Specifically, the key question we seek to answer is: find a k-way partitioning of the variables in the PGM where each of the first $k-1$ subsets is a block and the k-th subset contains all the collapsed variables, such that the estimation error is minimized and the resulting algorithm is tractable. This problem is non-trivial because of the complex interplay between collapsing and blocking. For example,

Example 1. Consider the pair-wise Markov network (undirected PGM) given in Fig. 1(a). Let us assume that each variable in the network has d values in its domain and our

[1]Collapsing is often called Rao-Blackwellisation. Technically, the latter is an advanced estimator while blocking and collapsing are advanced sampling strategies. In principle, we can also use the Rao-Blackwell estimator in blocked Gibbs sampling [4]. In this paper, we will separate sampling from estimation.



(a) A PGM (b) Collapsed PGM

Figure 1: Example to illustrate trade-off between blocking and collapsing.

time and memory resource constraints dictate that we cannot incur more than $O(d^3)$ complexity. Let us further assume that we have prior knowledge that A, B, C, and D should be blocked in order to improve the estimation accuracy (for instance, they are highly correlated or involved in deterministic constraints). Notice that we can only collapse (eliminate) E and F from the PGM . Otherwise, we will violate the complexity constraints. However, eliminating both E and F yields a clique over A, B, C, D (see Fig. 1(b)) and we can no longer block these variables because the complexity of computing a joint distribution over them (using junction tree propagation) and then sampling from it is $O(d^4)$. A much better solution in this case is to collapse F, create two blocks $\{A, B, C, D\}$ and $\{E\}$ and perform blocked Gibbs sampling over this sub-space.

As seen from the above example, in computation-limited settings, in many cases, variables that can be blocked in the original PGM can no longer be blocked in the collapsed PGM. In other words, there is a trade-off between blocking and collapsing which needs to be taken into account while combining the two schemes. We model this tradeoff by (i) defining two integer parameters α and β which bound the complexity of collapsing and blocking respectively and thus allow the user to control the number of blocked versus collapsed variables; (ii) defining two scoring functions, one each for blocking and collapsing, which favor blocks that contain variables that are highly correlated with each other and the collapsed set that contains variables which are highly correlated with other variables in the network; and (iii) casting the problem of finding the k-way partitioning into blocked and collapsed variables as a multi-objective optimization problem. This problem seeks to simultaneously maximize the scoring functions subject to the tractability constraints enforced by α and β.

The optimization problem is $\mathcal{NP}$-hard in general and therefore we propose a dynamic, greedy algorithm to solve it approximately. We integrate this algorithm with blocked-collapsed Gibbs sampling yielding a dynamic sampling algorithm. The algorithm begins by generating samples from the PGM using a feasible k-way partitioning computed using the (primal) graph associated with the PGM. It then periodically updates the partitioning after every M samples by leveraging the correlations computed from the generated samples and performs blocked-collapsed Gibbs sampling using the new partitioning. As more samples are drawn and as the accuracy of the measured correlation increases, the underlying Markov chain is likely to mix rapidly because highly correlated variables will be either blocked together or collapsed out.

We experimentally evaluate the efficacy of our new dynamic approach on several benchmark PGMs from literature. For comparison, we use (naive) Gibbs sampling, static blocked Gibbs sampling and static blocked collapsed Gibbs sampling. Our results show that on most of the benchmark PGMs, our dynamic approach is superior to the static graph-based blocked collapsed Gibbs sampling approaches.

The rest of the paper is organized as follows. In the next section, we present background. In section 3, we define the scoring functions and our optimization problem formulation. In section 4, we present a greedy approach to solve the optimization problem and describe our dynamic Gibbs sampling algorithm. In section 5, we present related work. Section 6 describes our experimental results and we conclude in section 7.

2 Background

In this section, we present our notation and provide a brief overview of PGMs, Gibbs sampling, blocking, collapsing and various estimation techniques. For details, see [5, 6, 7].

A (discrete) PGM or a Markov network, denoted by $\mathcal{M}$, is a pair $\langle \mathbf{X}, \Phi \rangle$ where $\mathbf{X} = \{X_1, \ldots, X_n\}$ is a set of discrete variables (i.e., they take values from a finite domain) and $\Phi = \{\phi_1, \ldots, \phi_m\}$ is a set of positive real-valued functions (or potentials). $\mathcal{M}$ represents the probability distribution $P(\overline{\mathbf{x}}) = \frac{1}{Z} \prod_{\phi \in \Phi} \phi(\overline{\mathbf{x}}_{S(\phi)})$ where $\overline{\mathbf{x}}$ is an assignment of values to all variables in $\mathbf{X}$, $\overline{\mathbf{x}}_{S(\phi)}$ is the projection of $\overline{\mathbf{x}}$ on the scope $S(\phi)$ of ϕ, and Z is a normalization constant called the partition function. We will often abuse notation and write $\phi(\overline{\mathbf{x}}_{S(\phi)})$ as $\phi(\overline{\mathbf{x}})$. The key inference tasks in PGMs are (i) computing the partition function; (ii) computing the 1-variable marginal probabilities, i.e., computing $P(\overline{x})$ where $\overline{x}$ is an assignment of a value in the domain of $X \in \mathbf{X}$ to X; and (iii) computing the most probable assignment, i.e., computing $\arg\max_{\overline{\mathbf{x}}} P(\overline{\mathbf{x}})$. In this paper, we focus on the task of computing 1-variable marginals.

The primal (or interaction) graph associated with $\mathcal{M} = \langle \mathbf{X}, \Phi \rangle$, denoted by $\mathcal{G}$, is an undirected graph which has variables of $\mathcal{M}$ as its vertices and an edge between any two variables that are contained in the scope of a function $\phi \in \Phi$. The primal graph is useful because several exact inference algorithms (e.g., the junction tree algorithm [8], AND/OR graph search [9], variable (bucket) elimination [10, 11], etc.) are exponential in the *treewidth* of the primal graph and thus the primal graph can be used to quantify their complexity. The treewidth of a graph $\mathcal{G}$, denoted by $tw(\mathcal{G})$, equals the *minimum width* over all possible orderings of its vertices. The width of an ordering (either partial or full) $\pi = (X_1, \ldots, X_n)$ of a graph $\mathcal{G}$, denoted by $w(\pi, \mathcal{G})$, is the maximum degree of X_i in $\mathcal{G}_{i-1}$, where $\mathcal{G} = \mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_n$ is a sequence of graphs such that $\mathcal{G}_i$ is obtained from $\mathcal{G}_{i-1}$ by adding edges so as to make the neighbor set of X_i in $\mathcal{G}_{i-1}$ a clique, and then removing X_i from $\mathcal{G}_i$ (i.e., eliminating X_i from $\mathcal{G}_{i-1}$). For example, Fig. 1(b) shows the graph obtained by eliminating E and F from the graph given in Fig. 1(a). The width of the partial order (E, F) is 2 while the width of the total order (E, F, A, B, C, D) is 3.

Computing the treewidth of a graph is a $\mathcal{NP}$-complete problem [12]. Therefore, in practice, we often employ heuristic approaches such as the *min-fill* heuristic and *min-degree* heuristic to find an upper-bound on the treewidth. Hereafter, whenever we refer to the treewidth of a graph, we implicitly assume that we have access to a close upper-bound to the treewidth.

2.1 Gibbs Sampling

Given a PGM $\mathcal{M} = \langle \mathbf{X}, \Phi \rangle$, Gibbs sampling [3] begins by initializing all variables randomly, denoted by $\overline{\mathbf{x}}^{(0)}$. Then, at each iteration j, it randomly chooses a variable $X_i \in \mathbf{X}$ and samples a value $\overline{x}_i$ for it from the conditional distribution $P(X_i | \overline{\mathbf{x}}_{-i}^{(j-1)})$, where $\overline{\mathbf{x}}_{-i}^{(j-1)}$ denotes the projection of $\overline{\mathbf{x}}^{(j-1)}$ on all variables in the PGM other than X_i. The new sample is $\overline{\mathbf{x}}^{(j)} = (\overline{x}_i, \overline{\mathbf{x}}_{-i}^{(j-1)})$. The computation of the conditional distribution can be simplified by observing that in a PGM, a variable X_i is conditionally independent of all other variables given its neighbors (or its *Markov blanket*) denoted by $MB(X_i)$. Formally, $P(X_i | \overline{\mathbf{x}}_{-i}) = P(X_i | \overline{\mathbf{x}}_{MB(X_i)})$. The Gibbs sampling procedure just described is called random-scan Gibbs sampling in literature. Another variation is systematic-scan Gibbs sampling in which we draw samples along a particular ordering of variables. It is known that random-scan Gibbs sampling is statistically more efficient than systematic-scan Gibbs sampling (cf. [7]).

2.2 Blocking and Collapsing

Blocked/Blocking Gibbs sampling [1] is an advanced sampling strategy in which some variables are sampled jointly given assignments to other variables in the PGM. Let the variables of the PGM be partitioned into disjoint groups or *blocks*, denoted by $\mathbb{B} = \{\mathbf{B}_i\}_{i=1}^{k}$, where $\mathbf{B}_i \subseteq \mathbf{X}$ and $\cup_i \mathbf{B}_i = \mathbf{X}$. Then, starting with a random assignment $\overline{\mathbf{x}}^{(0)}$ to all variables in the PGM, in each iteration j of blocked Gibbs sampling, we create a new sample $\overline{\mathbf{x}}^{(j)}$ by replacing the assignment to all variables in a randomly selected block $\mathbf{B}_i$ in $\overline{\mathbf{x}}^{(j-1)}$ by a new assignment that is sampled (jointly) from the distribution, $P(\mathbf{B}_i | \overline{\mathbf{x}}_{\mathbf{X} \setminus \mathbf{B}_i}^{(j-1)})$, where $\overline{\mathbf{x}}_{\mathbf{X} \setminus \mathbf{B}_i}^{(j-1)}$ is the projection of $\overline{\mathbf{x}}^{(j-1)}$ on all variables not in $\mathbf{B}_i$. We define the Markov blanket (MB) of a block $\mathbf{B}_i$ as all other blocks that contain at least one variable in $MB(X_i)$, where $X_i \in \mathbf{B}_i$. Similar to Gibbs sampling, an assignment to all variables in $MB(\mathbf{B}_i)$ makes $\mathbf{B}_i$ conditionally independent of all other variables. Note that blocked Gibbs sampling is feasible only when every block $\mathbf{B}_i$ is tractable given an assignment to $MB(\mathbf{B}_i)$. These tractability constraints are often imposed in practice by putting a limit on the treewidth of the primal graph projected on the block.

Collapsing is an alternative technique for improving the accuracy of Gibbs sampling. Collapsing operates by eliminating or marginalizing out a subset of variables, say $\mathbf{C}$, from the PGM $\mathcal{M}$ yielding a collapsed PGM, $\mathcal{M}_{\mathbf{X} \setminus \mathbf{C}}$. Gibbs sampling is then performed on this smaller PGM and this improves its accuracy (because only a sub-space is sampled). In practice, collapsing is feasible only if there exists an order π of the variables in $\mathbf{C}$ such that the width of the ordering is bounded by a small constant.

2.3 Estimators

Given N samples $\{\overline{\mathbf{x}}^{(i)}\}_{i=1}^{N}$ drawn from the distribution P, we can use one of the following three estimators to compute the 1-variable marginals.

1. **Histogram estimator:**

$$\widehat{P}(\overline{x}_i) = \frac{1}{N} \sum_{j=1}^{N} \mathbb{I}_{\overline{x}_i}(\overline{\mathbf{x}}^{(j)})$$

   where $\mathbb{I}_{\overline{x}_i}(\overline{\mathbf{x}}^{(j)})$ is an indicator function which equals 1 if $\overline{x}_i$ appears in $\overline{\mathbf{x}}^{(j)}$ and 0 otherwise.

2. **Mixture Estimator:**

$$\widehat{P}(\overline{x}_i) = \frac{1}{N} \sum_{j=1}^{N} P(\overline{x}_i | \overline{\mathbf{x}}_{MB(X_i)}^{(j)})$$

3. **Rao-Blackwell Estimator:** This estimator generalizes the mixture estimator and is given by

$$\widehat{P}(\overline{x}_i) = \frac{1}{N} \sum_{j=1}^{N} P(\overline{x}_i | \overline{\mathbf{x}}_{\mathbf{R}}^{(j)})$$

   where $\mathbf{R} \subseteq \mathbf{X}$.

It has been shown that the Rao-Blackwell estimator has smaller variance than the mixture estimator which in turn has smaller variance than the histogram estimator [7] and thus the Rao-Blackwell and the mixture estimators should always be preferred. However, the Rao-Blackwell estimator requires more computation since we are essentially "ignoring" the samples on certain variables (non-sampled variables). The non-sampled variables, $\mathbf{X} \setminus \mathbf{R}$ should now be marginalized out to obtain the estimate $\widehat{P}(\overline{x}_i)$. Therefore, as the set of non-sampled variables grows larger, estimation becomes more accurate but also computationally more expensive.

All the three estimators can be used with blocked as well as collapsed Gibbs sampling. To use the Rao-Blackwell estimator with blocked Gibbs sampling, we simply find the block, say $\mathbf{B}$, in which the variable resides, set $\mathbf{R}$ equal to $\mathbf{X} \setminus \mathbf{B}$ and compute $P(\overline{x}_i|\overline{\mathbf{x}}_{\mathbf{R}}^{(j)})$ by marginalizing out all variables other than X_i in the block. These computations are tractable because the block is assumed to be tractable. In collapsed Gibbs sampling, we can use the Rao-Blackwell estimator to estimate the 1-variable marginals over all the collapsed variables.

3 Optimally Selecting Blocked and Collapsed Variables

Integrating blocking and collapsing is tricky because they interact with each other. Moreover, we cannot collapse and block indiscriminately because for our algorithm to be practical we need to ensure that both blocking and collapsing are computationally tractable. In order to capture these constraints and the complex interplay between blocking and collapsing in a principled manner, we formulate the problem of selecting the blocks and collapsed variables as an optimization problem, defined next.

Definition 1. Given a PGM $\mathcal{M} = \langle \mathbf{X}, \Phi \rangle$, two scoring functions ω and ψ for blocking and collapsing respectively (see sec. 3.1), and integer parameters α, and β, find a k-way partition of $\mathbf{X}$ denoted by $\mathbb{X} = \mathbb{B} \cup \mathbf{C}$, where $\mathbb{B} = \{\mathbf{B}_i\}_{i=1}^{k-1}$ is a set of $k-1$ blocks and $\mathbf{C}$ is the set of collapsed variables such that both $\omega(\mathbb{B})$ and $\psi(\mathbf{C})$ are maximized, subject to two tractability constraints: (i) The minimum width of $\mathbf{C}$ in the primal graph $\mathcal{G}$ is bounded by α; and (ii) The treewidth of $\mathcal{G}_{\setminus \mathbf{C}}$ (the graph obtained by eliminating $\mathbf{C}$ from $\mathcal{G}$) projected on each block $\mathbf{B}_i$ is bounded by β, namely, $\forall\, \mathbf{B}_i \in \mathbb{B}$, $tw(\mathcal{G}_{\setminus \mathbf{C}}(\mathbf{B}_i)) \leq \beta$.

The optimization problem just presented requires maximizing two functions and is thus an instance of a multi-objective optimization problem [13, 14]. As one can imagine, this problem is much harder than typical optimization problems in machine learning which require optimizing just one objective function. In general, there may not exist a feasible solution that simultaneously optimizes each objective function. Therefore, a reasonable approach is to find a *Pareto optimal solution* , i.e., a solution which is not dominated by any other solution in the solution space. A Pareto optimal solution cannot be improved with respect to any objective without worsening another objective.

To find Pareto optimal solutions, we will use the *lexicographic method* – a well-known approach for managing the complexity of multi-objective optimization problems. In this method, the objective functions are arranged in order of importance and we solve a sequence of single objective optimization problems. Since collapsing changes the structure of the primal graph while blocking does not, it is obvious that we should first find the collapsed variables (i.e., give more importance to the objective function for collapsing) and then compute the blocks. We will use this approach. To reduce the sensitivity of the final solution to the objective-function for collapsing, we introduce a hard penalty which penalizes solutions that result in small block sizes (since the accuracy typically increases with the block size). We describe our proposed scoring (objective) functions and the hard penalty used next.

3.1 Scoring Functions

We wish to design scoring functions such that they improve mixing time of the underlying Markov chain. Since the exact mixing time is hard to compute analytically, we use a heuristic scoring function that uses correlations between the variables measured periodically from the generated samples. In general, collapsing variables is much more effective when the collapsed variables exhibit high correlation with other variables in the PGM. For instance, a variable X that is involved in a deterministic dependency (or constraint) with another variable Y (e.g., $Y = y \rightarrow X = x$) is a good candidate for collapsing; sampling such variables likely causes the Markov chain to get stuck and hinders mixing. Similarly, blocking is effective when we jointly sample variables which are tightly correlated because sampling them separately may cause the sampler to get trapped. Moreover, we also want to minimize the number of blocks or maximize the number of variables in each block because sampling a variable jointly with other variables in a block is better than or at least as good as sampling the variables individually [7]. We quantify these desirable properties using the following scoring functions:

$$\omega(\mathbb{B}) = \frac{1}{|\mathbb{B}|} \sum_{\mathbf{B}_i \in \mathbb{B}} \sum_{X_j, X_k \in \mathbf{B}_i} D(X_j, X_k) \qquad (1)$$

where $D(X_i, X_j)$ is any distance measure between the joint distribution $P(X_i, X_j)$ and the product of the marginal distributions $P(X_i)P(X_j)$.

$$\psi(\mathbf{C}) = \sum_{i=1}^{p} \frac{1}{|\mathbf{X} \setminus \mathbf{C}_{i-1}|} \sum_{X \in \mathbf{X} \setminus \mathbf{C}_{i-1}} D(C_i, X) \qquad (2)$$

where $(C_1, \ldots, C_p)$ is a user-defined order on variables in $\mathbf{C}$, $\mathbf{C}_i = \{C_1, \ldots, C_i\}$ and $C_0 = \emptyset$. We use the Hellinger distance, which is a symmetric measure to compute $D(X_i, X_j)$. Formally, this distance is given by:

$$D(X_i, X_j) = \frac{1}{\sqrt{2}} \sqrt{\sum_{\overline{x}_i, \overline{x}_j} \left(\sqrt{P(\overline{x}_i, \overline{x}_j)} - \sqrt{P(\overline{x}_i)P(\overline{x}_j)} \right)^2}$$

Algorithm 1: Greedy-Collapse

```
Input: A PGM $\mathcal{M} = \langle \mathbf{X}, \Phi \rangle$, Integers $\alpha$, and $\gamma$
Output: The collapsed PGM $\mathcal{M}_{\mathbf{X}/\mathbf{C}}$ obtained by eliminating $\mathbf{C}$
        from $\mathcal{M}$
1 $E = 0$; $\mathbf{C} = \emptyset$;
2 repeat
      // Let $\mathcal{G}$ be the primal graph associated
         with $\mathcal{M}$
3     Compute the value of the heuristic evaluation function for
      each vertex in $\mathcal{G}$ (see Eq. (4) );
4     Select a variable $X$ with the maximum heuristic value such
      that the degree $deg(X, \mathcal{G}) \leq \alpha$ where $deg(X, \mathcal{G})$ is the
      degree of $X$ in $\mathcal{G}$ ;
      // Let $E(X, \mathcal{G})$ be the number of new edges
         added to $\mathcal{G}$ by forming a clique over
         neighbors of $X$
5     $E = E + E(X, \mathcal{G})$;
6     Eliminate $X$ from $\mathcal{M}$;
7     $\mathbf{C} = \mathbf{C} \cup \{X\}$;
8 until all vertices in $\mathcal{G}$ have degree larger than $\alpha$ or $E > \gamma$;
9 return $\mathcal{M}$;
```

$D(X_i, X_j)$ measures the statistical dependence (correlation) between variables. Higher values indicate that the variables are statistically dependent while smaller values indicate that the variables are statistically independent. Notice that in order to compute $D(C_i, C_j)$, we need to know the 1-variable and 2-variable marginals. Their exact values are clearly not available and therefore we propose to estimate them from the generated samples.

As mentioned above, since we choose the collapsed variables before constructing the blocks, we have to penalize the feasible solutions that are likely to yield small blocks. We impose this penalty by using a hard constraint. The hard constraint disallows all feasible solutions $\mathbf{C}$ such that eliminating all variables in $\mathbf{C}$ along the ordering $(C_1, \ldots, C_p)$ will add more than γ edges to the primal graph. Thus, γ controls the relative importance of blocking versus collapsing. When γ is infinite or sufficiently large, the optimal solution to the objective function for collapsing is further refined to construct the blocks. On the other hand, when γ is small, a suboptimal solution to the objective function for collapsing, which can in turn enable higher quality blocking, is refined to construct the blocks.

4 Dynamic Blocked-Collapsed Gibbs Sampling

Although splitting the multi-objective optimization problem into two single objective optimization problems makes it comparatively easier to handle, it turns out that the resulting single objective optimization problems are $\mathcal{NP}$-hard. For instance, the problem of computing the set of collapsed variables includes the $\mathcal{NP}$-hard problem of computing the (weighted) treewidth (cf. [12]) as a special case. We therefore solve them using greedy methods.

4.1 Solving the optimization problem for Collapsing

Our greedy approach for computing the collapsed variables is given in Alg. 1. The algorithm takes as input the PGM $\mathcal{M}$, two integer parameters α and γ which constrain the width of the collapsed variables (tractability constraints) and the total number of edges added to the primal graph after eliminating the collapsed variables (penalty) respectively, selects the collapsed variables, and outputs a PGM obtained by eliminating the collapsed variables.

Alg. 1 heuristically selects variables one by one for collapsing until no variables can be selected because they will violate either the tractability constraints or the (penalty) constraint on the total number of edges added. For maximizing the objective function, we want to collapse as many highly correlated variables as possible. Thus, a simple greedy approach would be to select, at each iteration, the variable X with the maximum correlation score $\psi(X)$ where $\psi(X)$ is given by

$$\psi(X) = \frac{1}{|\mathbf{X}|} \sum_{X_i \in \mathbf{X}} D(X, X_i) \tag{3}$$

However, this approach is problematic because a highly correlated variable may add several edges to the primal graph, potentially increasing its treewidth. This will in turn constrain future selections and may yield solutions which are far from optimal. In other words, at each iteration, we have to balance locally maximizing the scoring function with the number of edges added in order to have a better chance of hitting the optimum or getting close to it. We therefore use the following heuristic evaluation function to evaluate the various choices:

$$\chi(X) = \psi(X) + \left(\frac{\binom{\alpha}{2} - E(X, \mathcal{G})}{\binom{\alpha}{2}} \right) \tag{4}$$

where $\psi(X)$ is defined in Eq. (3) and $E(X, \mathcal{G})$ is the number of new edges that will be added to $\mathcal{G}$ by forming a clique over X. Note that since the maximum degree of any eliminated variable is bounded by α, the maximum number of edges that can be added is bounded by $\binom{\alpha}{2}$. Therefore, the quantity in the brackets in Eq. (4) lies between 0 and 1 and high values for this quantity are desirable since very few edges will be added by eliminating the particular variable ($\psi(X)$ also lies between 0 and 1 and high values for it are desirable too).

4.2 Solving the optimization problem for Blocking

Alg. 2 presents the pseudo-code for our greedy approach for constructing the blocks. The algorithm takes as input a PGM $\mathcal{M}$ and an integer parameter β which bounds the treewidth of the primal graph of $\mathcal{M}$ projected on each block, and outputs a partitioning of the variables of $\mathcal{M}$ into blocks. The algorithm begins by having $|\mathbf{X}|$ blocks, each containing just one variable. Then it greedily merges two

Algorithm 2: Greedy-Block

```
Input: A PGM $\mathcal{M} = \langle \mathbf{X}, \Phi \rangle$ and Integer $\beta$
Output: A partition of $\mathbf{X}$ denoted by $\mathbb{B}$
1 Initialize $\mathbb{B} = \{\{X\}|X \in \mathbf{X}\}$ (each block contains just one variable);
2 repeat
      // Let $\mathbb{B}_{i,j}$ denote the partitioning formed from $\mathbb{B}$ by merging two blocks $\mathbf{B}_i, \mathbf{B}_j$ in $\mathbb{B}$
3     Merge two blocks $\mathbf{B}_i$ and $\mathbf{B}_j$ in $\mathbb{B}$ such that:
        1. they are in the Markov blanket of each other,
        2. $tw(\mathcal{G}(\mathbf{B}_i \cup \mathbf{B}_j)) \leq \beta$
        3. there does not exist another pair $\mathbf{B}_k$, $\mathbf{B}_m$ in $\mathbb{B}$ which satisfies the above two constraints and $\omega(\mathbb{B}_{k,m}) > \omega(\mathbb{B}_{i,j})$
4 until $\forall \mathbf{B}_i, \mathbf{B}_j \in \mathbb{B}$, $tw(\mathcal{G}(\mathbf{B}_i \cup \mathbf{B}_j)) > \beta$;
5 return $\mathbb{B}$;
```

Algorithm 3: Dynamic Blocked-Collapsed Sampling

```
Input: A PGM $\mathcal{M} = \langle \mathbf{X}, \Phi \rangle$; integers $T$, $M$; integers $\alpha$, $\beta$ and $\gamma$
Output: An estimate of marginal probabilities for all $X \in \mathbf{X}$
1 Initialize all 1-variable $P(\overline{x}_i)$ and 2-variable marginals $P(\overline{x}_i, \overline{x}_j)$ to zero;
2 for $t$ = 1 to $T$ do
3     $\mathcal{M}_{\mathbf{X} \setminus \mathbf{C}}$ = Greedy-Collapse($\mathcal{M}$,$\alpha$,$\gamma$);
4     $\mathbb{B}$ = Greedy-Block($\mathcal{M}_{\mathbf{X} \setminus \mathbf{C}}$,$\beta$);
5     Generate $M$ samples from $\mathcal{M}_{\mathbf{X} \setminus \mathbf{C}}$ using Blocked Gibbs sampling with $\mathbb{B}$ as blocks;
6     Update all 1-variable $P(\overline{x}_i)$ and 2-variable marginals $P(\overline{x}_i, \overline{x}_j)$ using the Rao-Blackwell estimator (see Eq. (5)).
  return $P(\overline{x}_i)$ for all variable-value combinations.
```

blocks such that they will yield the maximum increase in the score $\omega(\mathbb{B})$ under the constraint that the treewidth of the merged block is bounded by β. (Note that computing the treewidth is $\mathcal{NP}$-hard [12] and therefore in our implementation we use the min-fill algorithm to compute an upper bound on it.) To guard against merging blocks which are far away from each other in the primal graph (and thus likely to be statistically independent), we merge two blocks only if they are in the Markov blanket of each other.

4.3 Dynamic Blocked Collapsed Gibbs sampling

Next, we describe how to use the greedy blocking and collapsing algorithms within a Gibbs sampler, yielding an advanced sampling technique. Our proposal is summarized in Alg. 3. The algorithm takes as input a PGM $\mathcal{M}$, parameters α, β and γ for performing blocking and collapsing, and two integers T and M which specify the sample size and the interval at which the statistics are updated. At termination, the algorithm outputs an estimate of all 1-variable marginal probabilities.

The algorithm maintains an estimate of 1-variable and 2-variable marginals. The 2-variable marginals are used for computing the scoring functions. At each iteration, given a k-way partitioning of the variables into blocked and collapsed variables, denoted by $\mathbb{B}$ and $\mathbf{C}$ respectively, the algorithm generates M samples via blocked Gibbs sampling over $\mathcal{M}_{\mathbf{X} \setminus \mathbf{C}}$. After every M samples the algorithm updates the blocks and collapsed variables using the greedy procedures outlined in the previous two subsections. The 1-variable and 2-variable marginals are updated using the Rao-Blackwell estimator.

Next, we describe how to update the 1-variable marginals (2-variable marginals can be updated analogously). At each iteration t where $t \in \{1, T\}$, let $\{\overline{\mathbf{x}}^{(i,t)}\}_{i=1}^{M}$ be the set of M samples generated via Blocked Gibbs sampling and let $\widehat{P}_t(\overline{x})$ denote the estimate of $P(X = x)$ at iteration t. Then $\widehat{P}_t(\overline{x})$ is given by:

$$\widehat{P}_t(\overline{x}) = \frac{(t-1)\widehat{P}_{t-1}(\overline{x}) + Q_t(\overline{x})}{t} \quad (5)$$

where $Q_t(\overline{x})$ is computed as follows. If $X \in \mathbf{C}$ is a collapsed variable, then without loss of generality, let $\mathbf{B}_k$ denote the largest block in $\mathbb{B}$. Similarly, If X is a blocked variable, then without loss of generality, let $\mathbf{B}_k$ denote the block in $\mathbb{B}$ in which X is present. Let $\overline{\mathbf{x}}_{-k}^{(i,t)}$ denote the projection of $\overline{\mathbf{x}}^{(i,t)}$ on all variables in $\mathbb{B} \setminus \mathbf{B}_k$. Then Q_t is given as follows:

$$Q_t(\overline{x}) = \frac{1}{M} \sum_{i=1}^{M} P(\overline{x}|\overline{\mathbf{x}}_{-k}^{(i,t)}) \quad (6)$$

To compute $P(\overline{x}|\overline{\mathbf{x}}_{-k}^{(i,t)})$ we have to marginalize out all variables in $\mathbf{B}_k \cup \mathbf{C} \setminus \{X\}$. Computing this is tractable because according to our assumptions marginalizing out $\mathbf{C}$ is tractable. After marginalizing out $\mathbf{C}$, marginalizing out $\mathbf{B}_k$ is tractable because its treewidth is bounded by β.

M controls the rate at which the blocks and collapsed variables are updated. Ideally, it should be greater than the burn-in period. Also, although we have assumed a constant M, it is easy to envision setting it using a policy in which M is progressively increased as t increases. From the theory of adaptive MCMC [15], it is easy to show that such a policy will ensure that estimates output by Alg. 3 will converge to $P(\overline{x}_i)$ as T tends to infinity.

Note that when the correlation statistics are not available, i.e., when $t = 0$, the blocked and collapsed variables are computed by consulting the primal graph of the PGM. Thus, the blocks are constructed by randomly merging variables which are in the Markov blanket of each other; ties broken randomly. Similarly, the collapsed variables are selected along a constrained min-fill ordering (constrained by α). Thus, if we use a time bound, namely we stop sampling after the time bound has expired, and set M to be sufficiently large, Alg. 3 is equivalent to a static graph-based blocked-collapsed Gibbs sampling procedure.

Algorithm	Blocked	Collapsed	RB	Dynamic?
Geman & Geman [3]	N	N	N	N
Jensen et al. [1]	1	N	N	N
Bidyuk & Dechter [16]	N	Y	Y	N
Hamze & de Freitas [4]	2	N	Y	N
Paskin [17]	M	Y	Y	N
Our work	M	Y	Y	Y

Figure 2: Table comparing our work with previous work. Blocking (1: uses a single block, 2: uses 2 blocks, M: uses multiple blocks, N: not blocked), collapsing (Y/N), Rao-Blackwell Estimation (RB) (Y/N) and Dynamic (N: Static,Y: Dynamic).

5 Related Work

A number of earlier papers have investigated blocking and collapsing in the context of PGMs. Fig. 2 summarizes some notable ones and how they are related to our work. Blocked Gibbs sampling was first proposed by Jensen et al. [1]. The key idea in their algorithm was to create a "single block" by removing variables one by one from the primal graph until the treewidth of the remaining network is bounded by a constant and then sample this block using the junction tree algorithm. Unlike Jensen et al.'s work, we allow multiple blocks, combine collapsing with blocking and use the Rao-Blackwell estimator for computing the marginals (Jensen et al. use the histogram estimator).

Our algorithm is related to the Rao-Blackwellised blocked Gibbs sampling (RBBG) algorithm proposed by Hamze and de Freitas [4]. RBBG operates by dividing the network into two tractable tree-structured blocks and then performing Rao-Blackwellised estimation in each block. Unlike our algorithm, RBBG is applicable to grid Markov networks only. Also, unlike our algorithm, RBBG does not use multiple blocks and does not update the blocks dynamically. Moreover, RBBG does not use collapsing.

Another related work is that of Bidyuk and Dechter [16] in which the authors propose a collapsed Gibbs sampling algorithm. The key idea in their work is similar to Jensen et al.: remove variables one by one until the treewidth is bounded by a constant w (the removed variables form a w-cutset). However, unlike Jensen et al., they use the junction tree to sample the w-cutset variables. Formally, let $\mathbf{W}$ be the set of w-cutset variables and $\mathbf{V} = \mathbf{X} \setminus \mathbf{W}$ be the set of remaining variables. Then, the junction tree is used to compute the distribution $P(W_i|\mathbf{w}_{-i})$ and sample from it. Effectively, the set $\mathbf{V}$ is always collapsed out. A key drawback of this algorithm is that the junction tree algorithm must be run from scratch for sampling each w-cutset variable and as a result the algorithm can be quite slow. In this paper, we save time by marginalizing out a subset of variables before running the junction tree algorithm (i.e., marginalization is a pre-processing step before sampling). Also, unlike our work, the Bidyuk and Dechter algorithm does not use blocking and is not dynamic.

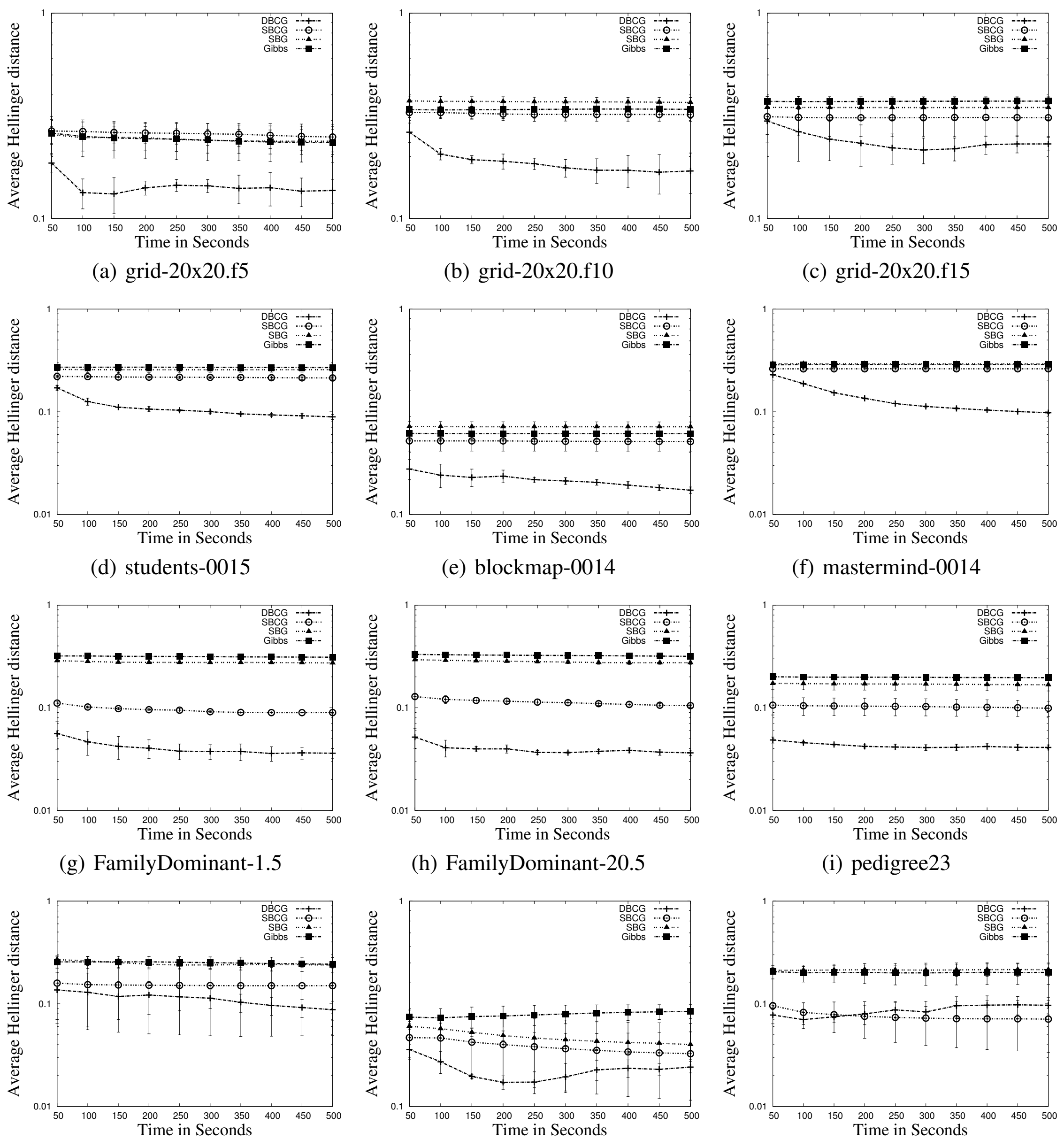
The sample propagation algorithm of Mark Paskin [17] is the only blocked-collapsed algorithm for PGMs that we are aware of. The algorithm integrates sampling with message passing in a junction tree. The key idea is to walk the clusters of a junction tree, sampling some of the current cluster's variables and then passing a message to one of its neighbors. The algorithm designates a subset of variables for sampling and marginalizes out the remaining variables by performing message passing over the junction. In that sense, sample propagation is similar to (but more efficient than) Bidyuk and Dechter's algorithm. The only difference is that variables within each cluster are sampled jointly (or blocked) if the cluster size is small enough or sampled using Metropolis-Hastings otherwise. Since the blocks in sample propagation are confined to the clusters of a junction tree, they can be much smaller than the blocks used in our algorithm. Also, this algorithm is not dynamic.

Our work is related to the recent work of Venugopal and Gogate [18], who cast the problem of constructing blocks in lifted Gibbs sampling as an optimization problem, but do not update the blocks dynamically. Finally, our work is related to parallel Gibbs sampling by Gonzalez et al. [19] who use likelihood estimates to compute the blocks.

6 Experiments

In this section, we experimentally evaluate the performance of the following algorithms on several benchmark PGMs: (*a*) Naive Gibbs sampling (`Gibbs`); (*b*) Static Blocked Gibbs sampling (`SBG`); (*c*) Static blocked collapsed Gibbs sampling (`SBCG`); and (*d*) Dynamic blocked collapsed Gibbs sampling (`DBCG`) . `SBG` is similar to the algorithm of Hamze and de Freitas [4] except that we allow multiple blocks and do not constrain the blocks to be tree structured. `SBCG` is an advanced version of Paskin's sample propagation algorithm [17]. We implemented `SBG` and `SBCG` by setting M to a sufficiently large value i.e., these methods consult only the primal graph of the PGM to choose the blocks and collapsed variables. To compute marginals, we use the Rao-Blackwell estimator in `SBG`, `SBCG` and `DBCG`, and the mixture estimator in `Gibbs`.

We tested the algorithms on several benchmark PGMs used in the UAI-2008 (graphmod.ics.uci.edu/uai08/Evaluation/Report), and the UAI-2010 (cs.huji.ac.il/project/PASCAL) probabilistic inference competitions. For each network, we measured performance using the average Hellinger distance between the true 1-variable marginals and the estimated 1-variable marginals. We performed our experiments on a centOS machine with a quad-core processor and 8GB RAM. Each algorithm was run for 500 seconds on each benchmark for the task of estimating 1-variable marginals. In `DBCG` we set $\alpha = \beta = 8$, $\gamma = 50 \times \alpha$ and $M = 1000$. We evaluate the impact of α, β and γ in the next sub-section.



(a) grid-20x20.f5 (b) grid-20x20.f10 (c) grid-20x20.f15

(d) students-0015 (e) blockmap-0014 (f) mastermind-0014

(g) FamilyDominant-1.5 (h) FamilyDominant-20.5 (i) pedigree23

(j) or-chain-62 (k) or-chain-129 (l) or-chain-236

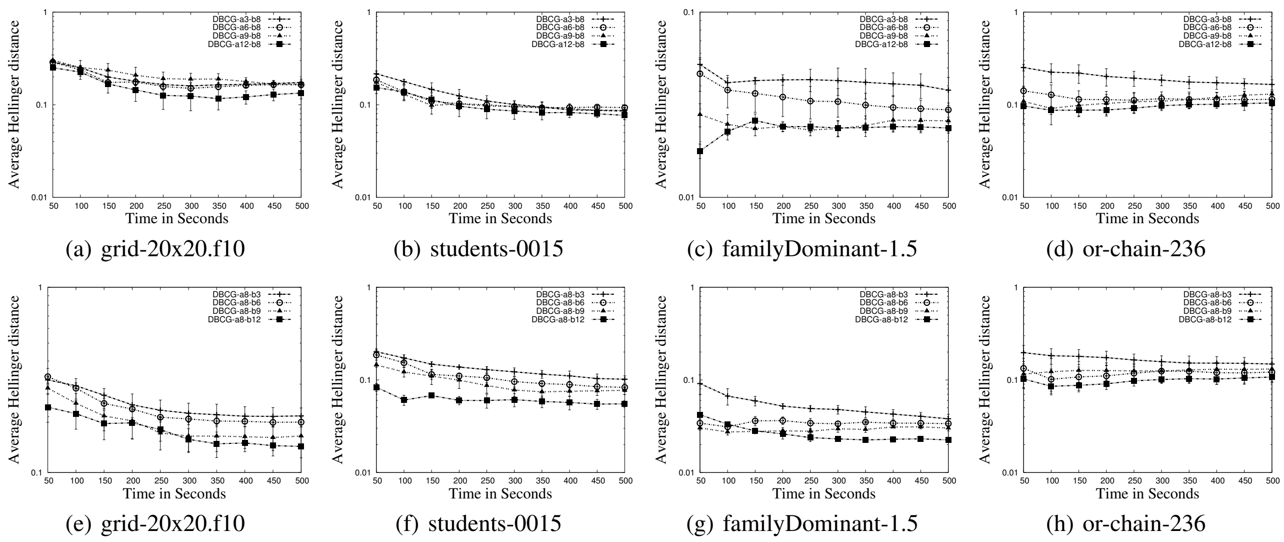
Figure 3: Average Hellinger distance between the exact and the approximate 1-variable marginals plotted as a function of time. (a)-(c): Grids, (d)-(g): Relational, (h)-(i): Linkage, (j)-(l): Promedas.

Fig. 3 shows the results. We see that DBCG is more accurate than all other algorithms on almost all the PGMs, often outperforming the competition by an order of magnitude.

Ising models. Figs. 3(a)-(c) show the performance of various algorithms on three Ising models of size 20×20 with evidence on 5, 10 and 15 randomly selected nodes respectively. DBCG is the best algorithm on all three PGMs. SBCG performs better than the other two algorithms on grid20x20.f10 and grid20x20.f15 and its performance is almost similar to SBG and Gibbs on grid20x20.f5.

Relational PGMs are formed by grounding statistical relational models [20]. Statistical relational models such as Markov logic networks [21, 22] often have large number of correlated variables as well as deterministic dependencies. Our dynamic approach is beneficial on such models because it has the ability to learn correlations and adjust the partitions accordingly. We experimented with three relational PGMs available from the UAI-08 repository: students-0015, blockmap-0014 and mastermind-0014. Figs. 3 (d)-(f) show the results. Again, we see that DBCG is the best performer followed by SBCG.

Linkage PGMs are used for performing genetic linkage analysis [23]. Figs. 3 (g)-(i) show results on three linkage PGMs. Again, on all three PGMs, DBCG is the best



(a) grid-20x20.f10 (b) students-0015 (c) familyDominant-1.5 (d) or-chain-236

(e) grid-20x20.f10 (f) students-0015 (g) familyDominant-1.5 (h) or-chain-236

Figure 4: Blocking vs. Collapsing tradeoff. (a)-(d): Impact of varying α with β set to a constant value. (e)-(f): Impact of varying β with α set to a constant value. We use $\gamma = 50 \times \alpha$. In all the plots, we plot the average Hellinger distance between the exact and the approximate 1-variable marginals as a function of time. The notation shown in the plots is as follows. DBCG-ax-by indicates that $\alpha = x$ and $\beta = y$.

performing algorithm and SBCG is the second best.

Promedas PGMs are noisy-OR medical diagnosis networks generated by the Promedas medical diagnosis system [24]. The networks are two-layered bi-partite graphs in which bottom layer has the symptoms and the top layer has the diseases. We experimented with three PGMs: or-chain-62, or-chain-129 and or-chain-236. Figs. 3 (j)-(l) show the results. DBCG performs better than all other algorithms in two out of the three PGMs. On or-chain-236, SBCG is slightly better than DBCG, but has larger variance.

6.1 Impact of varying the parameters α and β

Fig. 4 shows the impact of changing the parameters α and β on the performance of DBCG. For brevity, we show results on only one problem instance from each domain. Figs. 4 (a)-(d) show the impact of increasing α with β set to a constant while Figs. 4 (e)-(h) show the impact of increasing β with α set to a constant. We see that increasing α or β typically increases the accuracy and reduces the variance as a function of time. However, in some cases (e.g., Fig. 4(c) and Fig. 4(g)), we see that the accuracy goes down as we increase α and β, which indicates that there is a tradeoff between blocking and collapsing. In summary, α and β help us explore the region between a completely collapsed and a completely blocked sampler, and in turn help us achieve the right balance between blocking and collapsing.

7 Summary

In this paper, we formulated the problem of combining blocking and collapsing in computation-limited settings as a multi-objective optimization problem. We proposed a greedy algorithm to solve this problem. The greedy algorithm assumes access to correlations between all pairs of variables. Since the exact value of these correlations is not available, we proposed to estimate them from the generated samples, and update the greedy solution periodically. This yields a dynamic blocked collapsed Gibbs sampling algorithm which iterates between two steps: partitioning and sampling. In the partitioning step, the algorithm uses the current estimate of correlations between variables to partition the variables in the PGM into blocked and collapsed subsets and constructs the collapsed PGM. In the sampling step, the algorithm uses the blocks constructed in the previous step to generate samples from the collapsed PGM and updates the estimate of the 1-variable marginals and the correlations between variables. We performed a preliminary experimental study comparing the performance of our dynamic algorithm with static graph-based blocked collapsed Gibbs sampling algorithms. Our results clearly demonstrated the power and promise of our new approach: in many cases, our dynamic algorithm was an order of magnitude better in terms of accuracy than static graph-based algorithms.

Acknowledgements

This research was partly funded by the ARO MURI grant W911NF-08-1-0242. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of ARO or the U.S. Government.

References

[1] C. S. Jensen, U. Kjaerulff, and A. Kong, "Blocking Gibbs Sampling in Very Large Probabilistic Expert Systems," *International Journal of Human Computer Studies. Special Issue on Real-World Applications of Uncertain Reasoning*, vol. 42, pp. 647–666, 1993.

[2] J. S. Liu, W. H. Wong, and A. Kong, "Covariance structure of the Gibbs sampler with applications to the comparison of estimators and augmentation schemes," *Biometrika*, vol. 81, pp. 27–40, 1994.

[3] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[4] F. Hamze and N. de Freitas, "From Fields to Trees," in *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence*, pp. 243–250, 2004.

[5] A. Darwiche, *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

[6] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[7] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*. Springer Publishing Company, Incorporated, 2001.

[8] S. L. Lauritzen and D. J. Spiegelhalter, "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 50, no. 2, pp. 157–224, 1988.

[9] R. Dechter and R. Mateescu, "AND/OR Search Spaces for Graphical Models," *Artificial Intelligence*, vol. 171, no. 2-3, pp. 73–106, 2007.

[10] N. Zhang and D. Poole, "A simple approach to Bayesian network computations," in *Proceedings of the Tenth Biennial Canadian Artificial Intelligence Conference*, 1994.

[11] R. Dechter, "Bucket elimination: A unifying framework for reasoning," *Artificial Intelligence*, vol. 113, pp. 41–85, 1999.

[12] S. Arnborg, D. G. Corneil, and A. Proskurowski, "Complexity of finding embeddings in a k-tree," *SIAM Journal of Algebraic Discrete Methods*, vol. 8, pp. 277–284, Apr. 1987.

[13] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, pp. 369–395, April 2004.

[14] C. Hwang and M. A. S., *Multiple objective decision making, methods and applications: a state-of-the-art survey*. Springer-Verlag, 1979.

[15] G. Roberts and J. Rosenthal, "Coupling and Ergodicity of Adaptive MCMC," *Journal of Applied Probability*, vol. 44(2), pp. 458–477, 2007.

[16] B. Bidyuk and R. Dechter, "Cutset Sampling for Bayesian Networks.," *Journal of Artificial Intelligence Research*, vol. 28, pp. 1–48, 2007.

[17] M. A. Paskin, "Sample Propagation," in *Advances in Neural Information Processing Systems*, pp. 425–432, 2003.

[18] D. Venugopal and V. Gogate, "On lifting the gibbs sampling algorithm," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 1664–1672, 2012.

[19] J. Gonzalez, Y. Low, A. Gretton, and C. Guestrin, "Parallel gibbs sampling: From colored fields to thin junction trees," in *In Artificial Intelligence and Statistics*, May 2011.

[20] L. Getoor and B. Taskar, eds., *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[21] M. Richardson and P. Domingos, "Markov Logic Networks," *Machine Learning*, vol. 62, pp. 107–136, 2006.

[22] P. Domingos and D. Lowd, *Markov Logic: An Interface Layer for Artificial Intelligence*. San Rafael, CA: Morgan & Claypool, 2009.

[23] M. Fishelson and D. Geiger, "Optimizing Exact Genetic Linkage Computations," *Journal of Computational Biology*, vol. 11, no. 2/3, pp. 263–275, 2004.

[24] B. Wemmenhove, J. M. Mooij, W.Wiegerinck, M. A. R. Leisink, H. J. Kappen, and J. P. Neijt, "Inference in the promedas medical expert system," in *Eleventh Conference on Artificial Intelligence in Medicine*, pp. 456–460, 2007.

Bounded Approximate Symbolic Dynamic Programming for Hybrid MDPs

Luis Gustavo Rocha Vianna
University of Sao Paulo
Sao Paulo, Brazil
ludygrv@ime.usp.br

Scott Sanner
NICTA & ANU
Canberra, Australia
ssanner@nicta.com.au

Leliane Nunes de Barros
University of Sao Paulo
Sao Paulo, Brazil
leliane@ime.usp.br

Abstract

Recent advances in symbolic dynamic programming (SDP) combined with the extended algebraic decision diagram (XADD) data structure have provided exact solutions for mixed discrete and continuous (hybrid) MDPs with piecewise linear dynamics and continuous actions. Since XADD-based exact solutions may grow intractably large for many problems, we propose a bounded error compression technique for XADDs that involves the solution of a constrained bilinear saddle point problem. Fortuitously, we show that given the special structure of this problem, it can be expressed as a bilevel linear programming problem and solved to optimality in finite time via constraint generation, despite having an infinite set of constraints. This solution permits the use of efficient linear program solvers for XADD compression and enables a novel class of bounded approximate SDP algorithms for hybrid MDPs that empirically offers order-of-magnitude speedups over the exact solution in exchange for a small approximation error.

1 Introduction

Many real-world sequential-decision making problems involving resources, time, or spatial configurations naturally use continuous variables in both their state and action representation and can be modeled as Hybrid Markov Decision Processes (HMDPs). While HMDPs have been studied extensively in the AI literature [4; 7; 10; 9; 11; 12], only recently have symbolic dynamic programming (SDP) [14; 17] techniques been introduced to enable the exact solution of multivariate HMDPs with continuous actions and arbitrary piecewise linear dynamics and rewards.

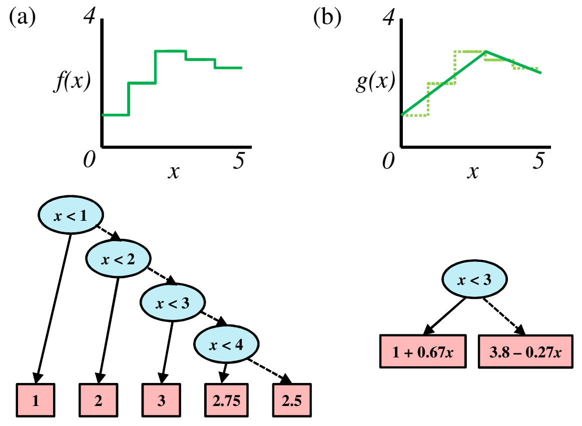


Figure 1: (a) A function $f(x)$ for $x \in [0, 5]$ and its XADD representation (solid branch is true, dotted branch is false); (b) A compressed XADD approximation $g(x)$ of $f(x)$. While these simple XADDs are trees, XADDs are more generally directed acyclic graphs as we show later.

What has proved crucial in this SDP solution of piecewise linear HMDPs is the use of the XADD data structure representation of functions like the simple examples shown in Figure 1(a,b) that allows the HMDP value function to be represented compactly and SDP operations to be computed efficiently. In brief, an XADD is simply an extension of the algebraic decision diagram (ADD) [1] to continuous variables where decisions may be boolean variable tests or inequalities of continuous expressions and leaves may be continuous expressions; XADDs are evaluated from root to leaf like decision trees. Following the SDP work of [17] for HMDPs with continuous actions that we extend, we restrict XADDs to have linear decisions and leaves.

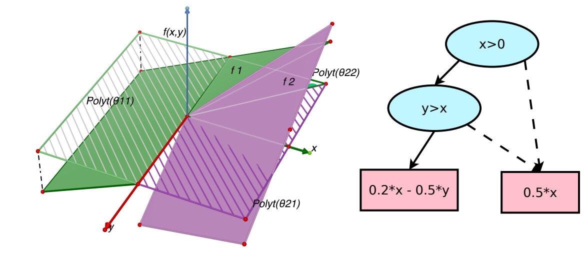
While XADDs have enabled SDP solutions to HMDPs that would not be otherwise possible with more naïve representations of piecewise functions, XADDs still have limitations — for some problems the HMDP solution (represented by a value function) simply has many distinct pieces and does not admit a more compact *exact* XADD representation, e.g, Figure 1(a). However, motivated by previous approximation work in discrete factored MDPs using ADD approximation [16], we pose the question of whether there exists a method for compressing an XADD in exchange for some bounded approximation error. As a hint of the solution, we note that Figure 1(a) can be approximated by 1(b) which is more compact and induces relatively small error.

But how do we find such a compressed XADD? In the simpler case of ADDs [16], this approximation process was straightforward: leaves with nearby constant values are averaged and merged, leading to bottom-up compaction of the ADD. In the XADD, if we wish to take a similar approach, we see that the problem is more complex since it is not clear (1) which leaves to merge, or (2) how to find the best approximation of these leaves that minimizes the error over the *constrained, continuous* space where each leaf is valid. Indeed, as Figure 1(a,b) demonstrates, the answer is *not* given simply by averaging leaves since the average of constant leaves in (a) could never produce the linear function in the leaves of (b). Hence, we wish to answer questions (1) and (2) to produce a *bounded and low-error approximation over the entire continuous function domain* as given in Figure 1(b).

To answer these questions, we propose a bounded error compression technique for linear XADDs that involves the solution of a constrained bilinear saddle point problem. Fortuitously, we show that given the special structure of this problem, it can be expressed as a bilevel linear programming problem. While the second-level optimization problem in this bilevel program implicitly represents an infinite number of constraints, we show that a constraint generation approach for this second stage allows the first stage to terminate at optimality after generating only a finite number of constraints. This solution permits the use of efficient linear program solvers for XADD compression and enables a novel class of bounded approximate SDP algorithms for hybrid MDPs. Empirically we demonstrate that this approach to XADD compression offers order-of-magnitude speedups over the exact solution in exchange for a small approximation error, thus vastly expanding the range of HMDPs for which solutions with strong error guarantees are possible.

2 Extended Algebraic Decision Diagrams (XADDs)

We begin with a brief introduction to the extended algebraic decision diagram (XADD), then in Section 3 we contribute approximation techniques for XADDs. In Section 4, we will show how this approximation can be used in a bounded approximate symbolic dynamic programming algorithm for hybrid MDPs.



$$f(x,y) = \begin{cases} \phi_1 : f_1 \\ \phi_2 : f_2 \end{cases} \quad \begin{aligned} \phi_1 &= \theta_{11} \vee \theta_{12} \\ \theta_{11} &= x < 0 \\ \theta_{12} &= x > 0 \wedge x < -y \\ f_1 &= \frac{x}{2} \end{aligned} \quad \begin{aligned} \phi_2 &= \theta_{21} \\ \theta_{21} &= x > 0 \wedge y > x \\ f_2 &= \frac{x}{5} - \frac{y}{2} \end{aligned}$$

Figure 2: Example of piecewise linear function in case and XADD form: *(top left)* plot of function $f(x,y)$; *(top right)* XADD representing $f(x,y)$; *(bottom)* case semantics for $f(x,y)$ demonstrating notation used in this work.

2.1 Case Semantics of XADDs

An XADD is a function represented by a directed acyclic graph having a fixed ordering of decision tests from root to leaf. For example, Figure 2(top left) shows the plot of a piecewise function and 2(top right) its XADD representation. Underlying this XADD is a simple piecewise linear function that we can represent semantically in case form. Specifically, given a domain of boolean and continuous variables $(\boldsymbol{b}^T, \boldsymbol{x}^T) = (b_1, \ldots, b_n, x_1, \ldots, x_m)$, where $b_i \in \{0,1\}$ $(1 \le i \le n)$ and $x_j \in [x_j^{\min}, x_j^{\max}]$ $(1 \le j \le m)$ for $x_j^{\min}, x_j^{\max} \in \mathbb{R}$ and $x_j^{\max} > x_j^{\min}$, a case statement representing an XADD with linear decisions and leaf expressions takes the following piecewise linear form

$$f(\boldsymbol{b},\boldsymbol{x}) = \begin{cases} \phi_1(\boldsymbol{b},\boldsymbol{x}) : & f_1(\boldsymbol{x}) \\ \vdots & \vdots \\ \phi_k(\boldsymbol{b},\boldsymbol{x}) : & f_k(\boldsymbol{x}) \end{cases} . \qquad (1)$$

Here the f_i are linear expressions over $\boldsymbol{x}$ and the ϕ_i are logical formulae defined over the domain $(\boldsymbol{b}^T, \boldsymbol{x}^T)$ that can include arbitrary logical $(\wedge, \vee, \neg)$ combinations of (i) boolean variables and (ii) linear inequalities over $\boldsymbol{x}$.

In the XADD example of Figure 2, *every leaf* represents a case value f_i and *every path from root to leaf* represents a conjunction of decision constraints. The disjunction of all path constraints leading to leaf f_i corresponds to a case partition $\langle \phi_i(\boldsymbol{b},\boldsymbol{x}) : f_i(\boldsymbol{x}) \rangle$. Clearly, all case partitions derived from an XADD must be mutually disjoint and exhaustive of the domain $(\boldsymbol{b}^T, \boldsymbol{x}^T)$, hence XADDs represent well-defined functions.

Since ϕ_i can be written in disjunctive normal form (DNF), i.e., $\phi_i \equiv \bigvee_{j=0}^{n_i} \theta_{ij}$ where θ_{ij} represents a conjunction of linear constraints over $\boldsymbol{x}$ and a (partial) truth assignment to $\boldsymbol{b}$ corresponding to the jth path from the XADD root to leaf f_i, we observe that every θ_{ij} contains a bounded convex linear polytope ($\boldsymbol{x}$ is finitely bounded in all dimensions as initially defined). We formally define the set of all convex linear polytopes contained in ϕ_i as $C(\phi_i) = \{Polytope(\theta_{ij})\}_j$, where *Polytope* extracts the subset of linear constraints from θ_{ij}. Figure 2(top left) illustrates the different polytopes in the XADD of 2(top right) with corresponding case notation in 2(bottom).

2.2 XADD Operations

XADDs are important not only because they compactly represent piecewise functions that arise in the forthcoming solution of hybrid MDPs, but also because operations on XADDs can efficiently exploit their structure. XADDs extend algebraic decision diagrams (ADDs) [1] and thus inherit most unary and binary ADD operations such as addition $\oplus$ and multiplication $\otimes$. While the addition of two linear piecewise functions represented by XADDs remains linear, in general their product may not (i.e., the values may be quadratic); however, for the purposes of symbolic dynamic programming later, we remark that we only ever need to multiply piecewise constant functions by piecewise linear functions represented as XADDs, thus yielding a piecewise linear result.

Some XADD operations do require extensions over the ADD, e.g., the binary max operation represented here in case form:

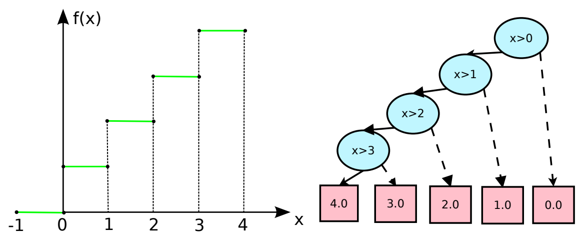
$$\max\left(\begin{cases}\phi_1 : f_1\\ \phi_2 : f_2\end{cases}, \begin{cases}\psi_1 : g_1\\ \psi_2 : g_2\end{cases}\right) = \begin{cases}\phi_1 \wedge \psi_1 \wedge f_1 > g_1 : f_1\\ \phi_1 \wedge \psi_1 \wedge f_1 \le g_1 : g_1\\ \phi_1 \wedge \psi_2 \wedge f_1 > g_2 : f_1\\ \phi_1 \wedge \psi_2 \wedge f_1 \le g_2 : g_2\\ \quad\vdots \qquad\qquad \vdots\end{cases}$$

While the max of two linear piecewise functions represented as XADDs remains in linear case form, we note that unlike ADDs which prohibit continuous variables $\boldsymbol{x}$ and *only* have a *fixed* set of boolean decision tests $\boldsymbol{b}$, an XADD may need to create *new decision tests* for the linear inequalities $\{f_1 > g_1, f_1 > g_2, f_2 > g_1, f_2 > g_2\}$ over $\boldsymbol{x}$ as a result of operations like max.

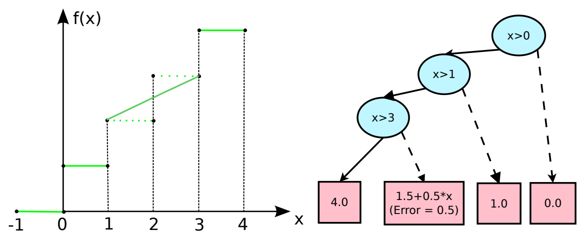
Additional XADD operations such as symbolic substitution, continuous (action) parameter maximization, and integration required for the solution of hybrid MDPs have all been defined previously [14; 17] and we refer the reader to those works for details.

3 Bounded XADD Approximation

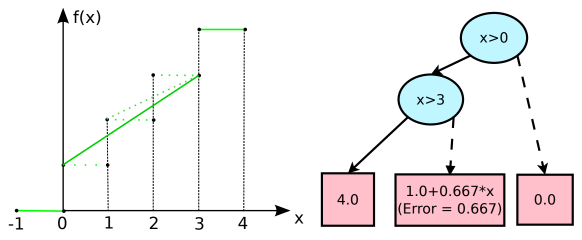
In this section, we present the main novel contribution of our paper for approximating XADDs within a fixed error bound. Since the point of XADD approximation is to shrink its size, we refer to our method of approximation as XADD Compression (`XADDComp`).



(a) Original



(b) Step1



(c) Step2

Figure 3: Successive pair merging XADD compression for a simple 1D example. At each step two nodes are chosen for merging, the best approximating hyperplane is determined according to Section 3.2, and if the accumulated error is within required bounds, the leaves are merged and internal XADD structure simplified to remove unneeded decisions.

Following previous work on ADD compression [16], we note that decision diagrams should be compressed from the bottom up — merging leaves causes simplifications to ripple upwards through a decision diagram removing vacuous decisions and shrinking the decision diagram. For example, after merging leaves in Figure 1(a), we note that the only remaining decision in 1(b) is $x < 3$. Hence, we focus on a leaf merging approach to `XADDComp`, which poses two questions: (1) what leaves do we merge? And (2) how do we find the best approximation of merged leaves? We answer these two questions in the following subsections.

3.1 Successive Leaf Merging

Since it would be combinatorially prohibitive to examine all possible leaf merges in an XADD, our `XADDComp` approximation approach in Algorithm 1 uses a systematic search strategy of successive pairwise merging of leaves. The idea is simple and is illustrated in Figure 3. The bounded error property is guaranteed by accumu-

Algorithm 1: $\texttt{XADDComp}(\text{XADD } X, \epsilon) \longrightarrow (\hat{X}, \hat{\epsilon})$

```
$\hat{\epsilon} \leftarrow 0$ // The max amount of error used so far
$\hat{X} \leftarrow X$ // The approximated XADD
$Open := \{L_i\} = \{\langle \phi_i, f_i \rangle\} \in \hat{X}$ // cases in $\hat{X}$
while $Open \neq \emptyset$ do
    $L_1 := Open.pop()$
    for $L_2 \in Open$ do
        // Merge and track accumulated error
        $(f^*, \tilde{\epsilon}) :=$ PairLeafApp$(L_1, L_2)$ // Sec 3.2
        $f^*.error := \tilde{\epsilon} + \max(f_1.error, f_2.error)$
        // Keep merge if within error bounds
        if $f^*.error < \epsilon$ then
            $\hat{\epsilon} := \max(\hat{\epsilon}, f^*.error)$
            $Open.remove(L_2)$
            // Replace leaves in $\hat{X}$ and simplify
            $\hat{X}.f_1 := f^*,\ \hat{X}.f_2 := f^*$
            $L_1 := \langle \phi_1 \vee \phi_2, f^* \rangle$ // Keep merging $L_1$
return $(\hat{X}, \hat{\epsilon})$ // Comp. XADD and error used
```

lating the amount of error "used" in every merged leaf and avoiding any merges that exceed a maximum error threshold ϵ. However, we have not yet defined how to find the lowest error approximation of a pair of leaves in `PairLeafApp`, which we provide next.

3.2 Pairwise Leaf Approximation

In pairwise leaf merging, we must address the following fundamental problem: given two XADD leaves represented by their case partitions $L_1 = \langle f_1, \phi_1 \rangle$ and $L_2 = \langle f_2, \phi_2 \rangle$, our goal is to determine the best linear case approximation of L_1 and L_2. As it must represent L_1 and L_2, the solution must be defined in both regions and is therefore of the form $L^* = \langle f^*, \phi_1 \vee \phi_2 \rangle$. Since we restrict to linear XADDs, then $f_1 = \boldsymbol{c_1}^T(\boldsymbol{x}^T, 1)^T$, $f_2 = \boldsymbol{c_2}^T(\boldsymbol{x}^T, 1)^T$ and $f^* = \boldsymbol{c^*}^T(\boldsymbol{x}^T, 1)^T$ (assuming $\boldsymbol{c_1}, \boldsymbol{c_2}, \boldsymbol{c^*} \in \mathbb{R}^{m+1}$ where $\boldsymbol{x} \in \mathbb{R}^m$). Thus, our task reduces to one of finding the optimal weight vector $\boldsymbol{c^*}$ which minimizes approximation error given by the following bilinear saddle point optimization problem:

$$\min_{\boldsymbol{c}^*} \max_{i \in \{1,2\}} \max_{\boldsymbol{x} \in C(\phi_i)} \left| \underbrace{\boldsymbol{c_i}^T \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix}}_{f_i} - \underbrace{\boldsymbol{c^*}^T \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix}}_{f^*} \right| \tag{2}$$

This is bilinear due to the inner product of $\boldsymbol{c^*}$ with $\boldsymbol{x}$.

To better understand the structure of this bilinear saddle point problem, we refer to Figure 4, which shows on the left, two functions f_1 and f_2 and the respective single polytope regions $C(\phi_1)$ and $C(\phi_2)$ where f_1 and f_2 are respectively valid. On the right, we show a proposed approximating hyperplane f within

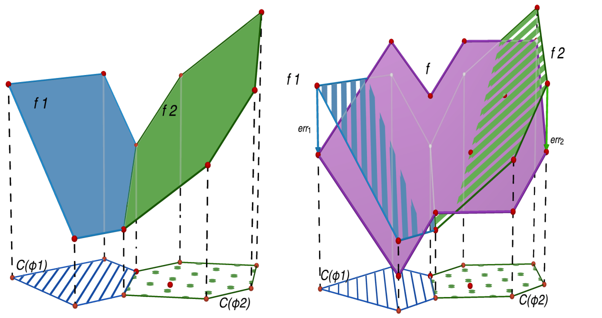


Figure 4: Illustration of the pairwise leaf approximation problem: *(left)* the original linear leaf functions f_1 and f_2 in their respective (single) polytope regions ϕ_1 and ϕ_2; *(right)* a linear approximation f overlaid on f_1 and f_2 in their regions showing errors at the polytope vertices.

the regions $C(\phi_1)$ and $C(\phi_2)$. Clearly we want to choose the $f^* = f$ that minimizes the absolute difference between f and f_1, f_2 within their respective polytopes. On account of this perspective and recalling that $C(\phi_i) = \{Polytope(\theta_{ij})\}_j$, we can rewrite (2) as the following bi-level linear optimization problem:[1]

$$\min_{\boldsymbol{c}^*, \epsilon} \epsilon \tag{3}$$
$$s.t.\ \epsilon \geq \begin{pmatrix} \max\limits_{\boldsymbol{x}} & \left| \boldsymbol{c_i}^T \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} - \boldsymbol{c^*}^T \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} \right| \\ s.t. & \boldsymbol{x} \in Polytope(\theta_{ij}) \end{pmatrix} ; \forall i \in \{1,2\}, \forall \theta_{ij}$$

While it may seem we have made little progress with this rewrite of (2) — this still appears to be a difficult optimization problem, we can make an important insight that allows us to remove the second stage of optimization altogether. While implicitly it appears that the second stage would correspond to an infinite number of constraints — one for each $\boldsymbol{x} \in Polytope(\theta_{ij})$, we return to Figure 4. Since each of f, f_1, and f_2 are all linear and $C(\phi_1), C(\phi_2)$ represent (unions of) linear convex polytopes, we know that the *maximum difference* between f and f_1, f_2 must occur at the *vertices* of the respective polytope regions. Thus, denoting $\boldsymbol{x_{ij}^k}$ $(k \in \{1 \ldots N_{ij}\})$ as a vertex of the linear convex polytope defined by θ_{ij}, we can obtain a *linear program* version of (2) with a *finite* number of constraints at the vertices $\boldsymbol{x_{ij}^k}$ of all polytopes:

$$\min_{\boldsymbol{c}^*, \epsilon} \epsilon \tag{4}$$
$$s.t.\ \epsilon \geq \left| \boldsymbol{c_i}^T \begin{bmatrix} \boldsymbol{x_{ij}^k} \\ 1 \end{bmatrix} - \boldsymbol{c^*}^T \begin{bmatrix} \boldsymbol{x_{ij}^k} \\ 1 \end{bmatrix} \right| ;\ \begin{matrix} \forall i \in \{1,2\}, \forall \theta_{ij}, \\ \forall k \in \{1 \ldots N_{ij}\} \end{matrix}$$

[1]To obtain a true bi-level *linear* program, we need *two* separate second stage constraints to encode that ϵ is larger than each side of the absolute value (the argument of the absolute value and its negation), but this is a straightforward absolute value expansion in a linear program that we will consider implicit to avoid notational clutter.

Algorithm 2: `PairLeafApp`$(L_1, L_2) \longrightarrow (\boldsymbol{c}^*, \epsilon)$

```
1  $\boldsymbol{c^*} := \boldsymbol{0}$ // Arbitrarily initialize $\boldsymbol{c^*}$
2  $\epsilon^* := \infty$ // Initialize to invalid error
3  $C := \emptyset$ // Start with an empty constraint set
4  // Generate max error vertex constraints for $\boldsymbol{c^*}$
5  for $i \in \{1,2\}, \theta_{ij} \in C(\phi_i)$ do
6      $\boldsymbol{x^k_{ij}}_+ := \arg\max_{\boldsymbol{x}} \left( \boldsymbol{c_i}^T \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} - \boldsymbol{c^*}^T \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} \right)$
           s.t. $\boldsymbol{x} \in Polytope(\theta_{ij})$
       $\boldsymbol{x^k_{ij}}_- := \arg\max_{\boldsymbol{x}} \left( \boldsymbol{c^*}^T \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} - \boldsymbol{c_i}^T \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} \right)$
           s.t. $\boldsymbol{x} \in Polytope(\theta_{ij})$
       $C := C \cup \{\epsilon > \boldsymbol{c_i}^T ({\boldsymbol{x^k_{ij}}_+}^T, 1)^T - \boldsymbol{c^*}^T ({\boldsymbol{x^k_{ij}}_+}^T, 1)^T\}$
7      $C := C \cup \{\epsilon > \boldsymbol{c^*}^T ({\boldsymbol{x^k_{ij}}_-}^T, 1)^T - \boldsymbol{c_i}^T ({\boldsymbol{x^k_{ij}}_-}^T, 1)^T\}$
8  // Re-solve LP with augmented constraint set
9  $(\boldsymbol{c^*}, \epsilon^*_{new}) := \arg\min_{\boldsymbol{c^*},\epsilon} \epsilon$ subject to $C$
10 if $\epsilon^*_{new} \neq \epsilon^*$ then
11     $\epsilon^* := \epsilon^*_{new}$, go to line 4
12 return $(\boldsymbol{c^*}, \epsilon^*)$// Best hyperplane and error
```

Unfortunately, the drawback of this single linear programming approach is that for M_{ij} linear constraints in *Polytope*(θ_{ij}), the number of vertices of the polytope may be *exponential*, i.e., $N_{ij} = O(\exp M_{ij})$.

However, we make one final crucial insight: while we may have an exponential number of constraints in (4), we have a very efficient way to evaluate for a *fixed* solution $\boldsymbol{c}^*$, the single point $\boldsymbol{x^k_{ij}}$ in each *Polytope*(θ_{ij}) with max error — this is exactly what the second stage linear program in (3) provides. Hence, this suggests an efficient *constraint generation* approach to solving (4) that we outline in Algorithm 2. Beginning with an empty constraint set, we iteratively add in constraints for the polytope vertices $\boldsymbol{x^k_{ij}}$ that yield maximal error for the current best solution $\boldsymbol{c^*}$ (one constraint for each side of the absolute value). Then we re-solve for $(\boldsymbol{c^*}, \epsilon^*)$ to see if the error has gotten worse; if not, we have reached optimality since $\boldsymbol{c^*}$ satisfies all constraints (vertices $\boldsymbol{x^k_{ij}}$ not having constraints in C had provably equal or smaller error than those in C) and adding in all constraints could not reduce ϵ^* further.

We conclude our discussion of `PairLeafApp` in Algorithm 2 with a key observation: it will always terminate in finite time with the optimal solution, since at least two constraints are generated on every iteration and there are only a finite number of possible polytope vertices $\boldsymbol{x^k_{ij}}$ for which to generate constraints. We later demonstrate that `PairLeafApp` runs very efficiently in practice indicating that it is generating only a small subset of the possible exponential set of constraints.

4 Bounded Approximate Symbolic Dynamic Programming

Having shown how to efficiently approximate XADDs in Section 3, we switch to the main application focus of this work: finding bounded approximate solutions for Hybrid MDPs (HMDPs). Specifically, in this section, we build on the Symbolic Dynamic Programming (SDP) [14; 17] framework for HMDPs that uses the XADD data structure to maintain a compact representation of the value function, extending it to allow next-state dependent rewards and synchronic arcs in its transition function. In this work, we augment SDP with a bounded value approximation step that we will subsequently show permits the solution of HMDPs with strong error guarantees that cannot be efficiently solved exactly. We begin by formalizing an HMDP.

4.1 Hybrid Markov Decision Processes (HMDPs)

In HMDPs, states are represented by variable assignments. We assume a vector of variables $(\boldsymbol{b}^T, \boldsymbol{x}^T) = (b_1, \ldots, b_n, x_1, \ldots, x_m)$, where each $b_i \in \{0,1\}$ $(1 \leq i \leq n)$ is boolean and each $x_j \in \mathbb{R}$ $(1 \leq j \leq m)$ is continuous. We also assume a finite set of p parametrized actions $A = \{a_1(\boldsymbol{y}_1), \ldots, a_p(\boldsymbol{y}_p)\}$, where $\boldsymbol{y}_k \in \mathbb{R}^{|\boldsymbol{y}_k|}$ $(1 \leq k \leq p)$ denote continuous parameters for respective action a_k (often we drop the subscript, e.g., $a(\boldsymbol{y})$).

An HMDP model also requires the following: (i) a joint state transition model $P(\boldsymbol{b}', \boldsymbol{x}' | \boldsymbol{b}, \boldsymbol{x}, a, \boldsymbol{y})$, which specifies the probability of the next state $(\boldsymbol{b}', \boldsymbol{x}')$ conditioned on a subset of the previous and next state and action $a(\boldsymbol{y})$; (ii) a reward function $R(\boldsymbol{b}, \boldsymbol{x}, a, \boldsymbol{y}, \boldsymbol{b}', \boldsymbol{x}')$, which specifies the immediate reward obtained by taking action $a(\boldsymbol{y})$ in state $(\boldsymbol{b}, \boldsymbol{x})$ and reaching state $(\boldsymbol{b}', \boldsymbol{x}')$; and (iii) a discount factor γ, $0 \leq \gamma \leq 1$.

A policy π specifies the action $a(\boldsymbol{y}) = \pi(\boldsymbol{b}, \boldsymbol{x})$ to take in each state $(\boldsymbol{b}, \boldsymbol{x})$. Our goal is to find an optimal sequence of finite horizon-dependent policies $\Pi^* = (\pi^{*,1}, \ldots, \pi^{*,H})$ that maximizes the expected sum of discounted rewards over a horizon $h \in H; H \geq 0$:

$$V^{\Pi^*}(\boldsymbol{b}, \boldsymbol{x}) = E_{\Pi^*} \left[\sum_{h=0}^{H} \gamma^h \cdot r^h \middle| (\boldsymbol{b}_0, \boldsymbol{x}_0) \right]. \qquad (5)$$

Here r^h is the reward obtained at horizon h following Π^* where we assume starting state $(\boldsymbol{b}_0, \boldsymbol{x}_0)$ at $h = 0$.

HMDPs as defined above are naturally factored [3] in terms of state variables $(\boldsymbol{b}, \boldsymbol{x})$; as such, transition structure can be exploited in the form of a dynamic Bayes net (DBN) [6] where the conditional probabilities $P(b'_i | \cdots)$ and $P(x'_j | \cdots)$ for each next state variable can condition on the action, current and next

Algorithm 3: BASDP(HMDP, H, ϵ) $\longrightarrow (V^h, \pi^{*,h})$

```
1  begin
2      $V^0 := 0, h := 0$
3      while $h < H$ do
4          $h := h + 1$
5          foreach $a \in A$ do
6              $Q_a^h(\boldsymbol{y}) :=$ Regress($V^{h-1}, a, \boldsymbol{y}$)
7              $Q_a^h := \max_{\boldsymbol{y}} Q_a^h(\boldsymbol{y})$ // Parameter max
8              $V^h := \max_a Q_a^h$ // Max all $Q_a$
9              $\pi^{*,h} := \arg\max_{(a,\boldsymbol{y})} Q_a^h(\boldsymbol{y})$
10         $V^h =$ XADDComp($V^h, \epsilon$)
11         if $V^h = V^{h-1}$ then
12             break // Stop if early convergence
13     return $(V^h, \pi^{*,h})$
14 end
```

Algorithm 4: Regress($V, a, \boldsymbol{y}$) $\longrightarrow Q$

```
1  begin
2      $Q$ = Prime($V$) // Rename all symbolic
       //variables $b_i \rightarrow b_i'$ and all $x_i \rightarrow x_i'$
3      $Q := R(\boldsymbol{b}, \boldsymbol{x}, a, \boldsymbol{y}, \boldsymbol{b}', \boldsymbol{x}') \oplus (\gamma \cdot Q)$
4      // Any var order with child before parent
5      foreach $v_k'$ in $Q$ do
6          if $v_k' = x_j'$ then
7              //Continuous marginal integration
8              $Q := \int Q \otimes P(x_j' | \boldsymbol{b}, \boldsymbol{x}, \boldsymbol{v}'_{<k}, a, \boldsymbol{y})\, d_{x_j'}$
9          if $v_k' = b_i'$ then
10             // Discrete marginal summation
11             $Q := \left[Q \otimes P(b_i' | \boldsymbol{b}, \boldsymbol{x}, \boldsymbol{v}'_{<k}, a, \boldsymbol{y})\right] |_{b_i'=1}$
                   $\oplus \left[Q \otimes P(b_i' | \boldsymbol{b}, \boldsymbol{x}, \boldsymbol{v}'_{<k}, a, \boldsymbol{y})\right] |_{b_i'=0}$
12     return $Q$
13 end
```

state. We allow *synchronic arcs* (variables that condition on each other in the same time slice) between any pair of variables, binary $\boldsymbol{b}$ or continuous $\boldsymbol{x}$ so long as they do not lead to cyclic dependencies in the DBN — this leads to a natural topologically sorted variable ordering that prevents any variable from conditioning on a later variable in the ordering. From these assumptions, we factorize the joint transition model as

$$P(\boldsymbol{b}', \boldsymbol{x}' | \boldsymbol{b}, \boldsymbol{x}, a, \boldsymbol{y}) = \prod_{k=1}^{n+m} P(v_k' | \boldsymbol{b}, \boldsymbol{x}, \boldsymbol{v}'_{<k}, a, \boldsymbol{y})$$

where $\boldsymbol{v}'_{<k} = (v_1', \ldots, v_{k-1}'), 1 \le k \le n + m$.

The conditional probability functions $P(b_i' = v_{k_i}' | \boldsymbol{b}, \boldsymbol{x}, \boldsymbol{v}'_{<k_i}, a, \boldsymbol{y})$ for *binary* variables b_i $(1 \le i \le n)$ can condition on state and action variables. For the *continuous* variables x_j $(1 \le j \le m)$, we represent the CPFs $P(x_j' = v_{k_j}' | \boldsymbol{b}, \boldsymbol{x}, \boldsymbol{v}'_{<k_j}, a, \boldsymbol{y})$ with *piecewise linear equations* (PLEs) satisfying three properties: (i) PLEs can only condition on the action, current state, and previous state variables, (ii) PLEs are deterministic meaning that to be represented by probabilities they must be encoded using Dirac $\delta[\cdot]$ functions and (iii) PLEs are piecewise linear, where the piecewise conditions may be arbitrary logical combinations of the binary variables and linear inequalities over the continuous variables. Numerous examples of PLEs will be presented in the empirical results in Section 5.

While it is clear that our restrictions do not permit general stochastic continuous transition noise (e.g., Gaussian noise), they do permit discrete noise in the sense that $P(x_j' = v_{k_j}' | \boldsymbol{b}, \boldsymbol{x}, \boldsymbol{v}'_{<k_j}, a, \boldsymbol{y})$ may condition on $\boldsymbol{b}'$, which are stochastically sampled according to their CPFs. We note that this representation effectively allows modeling of continuous variable transitions as a mixture of δ functions, which has been used frequently in previous exact continuous state MDP solutions [7; 12].

We allow the reward function $R(\boldsymbol{b}, \boldsymbol{x}, a, \boldsymbol{y}, \boldsymbol{b}', \boldsymbol{x}')$ to be a general piecewise linear function (boolean or linear conditions and linear values) such as

$$R(\boldsymbol{b}, \boldsymbol{x}, a, \boldsymbol{y}, \boldsymbol{b}', \boldsymbol{x}') = \begin{cases} b \wedge x_1 \le x_2 + 1 : & 1 - x_1' + 2x_2' \\ \neg b \vee x_1 > x_2 + 1 : & 3y + 2x_2 \end{cases}$$

The above transition and reward constraints ensure that all derived functions in the solution of these HMDPs will remain piecewise linear, which is essential for efficient linear XADD representation [14] and for the XADD approximation techniques proposed in Section 3.

4.2 Solution Methods

The algorithm we use for solving HMDPs is an approximate version of the continuous state and action generalization of *value iteration* [2], which is a dynamic programming algorithm for constructing optimal policies. It proceeds by constructing a series of h-stage-to-go optimal value functions $V^h(\boldsymbol{b}, \boldsymbol{x})$. Initializing $V^0(\boldsymbol{b}, \boldsymbol{x}) = 0$, we define the *quality* $Q_a^h(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{y})$ of taking action $a(\boldsymbol{y})$ in state $(\boldsymbol{b}, \boldsymbol{x})$ and acting so as to obtain $V^{h-1}(\boldsymbol{b}, \boldsymbol{x})$ thereafter as the following:

$$Q_a^h(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{y}) = \sum_{\boldsymbol{b}'} \int_{\boldsymbol{x}'} \left[\prod_{k=1}^{n+m} P(v_k' | \boldsymbol{b}, \boldsymbol{x}, \boldsymbol{v}'_{<k}, a, \boldsymbol{y}) \cdot \left(R(\boldsymbol{b}, \boldsymbol{x}, a, \boldsymbol{y}, \boldsymbol{b}', \boldsymbol{x}') + \gamma V^{h-1}(\boldsymbol{b}', \boldsymbol{x}') \right) \right] d\boldsymbol{x}' \quad (6)$$

Given $Q_a^h(\boldsymbol{b}, \boldsymbol{x})$ for each $a \in A$, we can proceed to define the h-stage-to-go value function as the maximizing action parameter values $\boldsymbol{y}$ for the best action a in each state $(\boldsymbol{b}, \boldsymbol{x})$ as follows:

$$V^h(\boldsymbol{b}, \boldsymbol{x}) = \max_{a \in A} \max_{\boldsymbol{y} \in \mathbb{R}^{|\boldsymbol{y}|}} \left\{ Q_a^h(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{y}) \right\} \qquad (7)$$

If the horizon H is finite, then the optimal value function is obtained by computing $V^H(\boldsymbol{b}, \boldsymbol{x})$ and the optimal horizon-dependent policy $\pi^{*,h}$ at each stage h can be easily determined via $\pi^{*,h}(\boldsymbol{b}, \boldsymbol{x}) = \arg\max_{(a,\boldsymbol{y})} Q_a^h(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{y})$. If the horizon $H = \infty$ and the optimal policy has finitely bounded value, then value iteration can terminate at horizon h if $V^h = V^{h-1}$; then $V^\infty = V^h$ and $\pi^{*,\infty} = \pi^{*,h}$.

4.3 Bounded Approximate SDP (BASDP)

We will now define BASDP, our bounded approximate HMDP symbolic dynamic programming algorithm. BASDP is provided in Algorithm 3 along with a regression subroutine in Algorithm 4; BASDP is a modified version of SDP [17] to support the HMDP model with next-state dependent reward function and synchronic arcs as defined previously along with the crucial addition of line 10, which uses the `XADDComp` compression method described in Section 3. Error is cumulative over each horizon, so for example, the maximum possible error incurred in an *undiscounted* BASDP solution is $H\epsilon$. All functions are represented as XADDs, and we note that all of the XADD operations involved, namely addition $\oplus$, multiplication $\otimes$, integration of Dirac δ functions, marginalization of boolean variables $\sum_{b_i}$, continuous parameter maximization $\max_{\boldsymbol{y}}$ and discrete maximization $\max_a$, are defined for XADDs as given by [14; 17]. For most of these operations the execution time scales superlinearly with the number of partitions in the XADD, which can be greatly reduced by the `XADDComp` compression algorithm. We empirically demonstrate the benefits of approximation in the next section.

5 Empirical Results

In this section we wish to compare the scalability of exact SDP (calling BASDP in Algorithm 4 with $\epsilon = 0$) vs. various levels of approximation error $\epsilon > 0$ to determine the trade-offs between time and space vs. approximation error. To do this, we evaluated BASDP on three different domains — Mars Rover1D, Mars Rover2Dand Inventory Control— detailed next.

Mars Rover1D: A unidimensional continuous Mars Rover domain motivated by Bresina *et al* [5] used in order to visualize the value function and the effects of varying levels of approximation. The position of the rover is represented by a single continuous variable x and the goal of the rover is to take pictures at specific positions. There is only one action $move(a_x)$, where

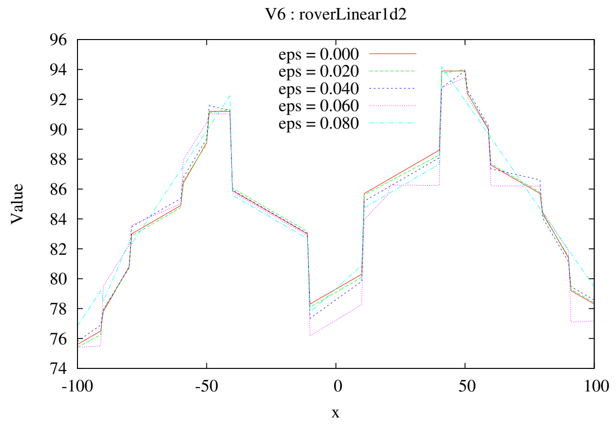


Figure 5: Value function at iteration 6 for Mars Rover1D, showing how different levels of approximation error (eps) lead to different compressions.

a_x is the movement distance. In the description of the problem for the instance shown below, there are two picture points and taking pictures is recorded in two boolean variables (tp_1 and tp_2). The dynamics for deterministic action $move(a_x)$ are as follows:

$$tp_1' = \begin{cases} tp_1 \vee (x > 40 \wedge x < 60) & : 1.0 \\ else & : 0.0 \end{cases}$$

$$tp_2' = \begin{cases} tp_2 \vee (x > -60 \wedge x < -40) & : 1.0 \\ else & : 0.0 \end{cases}$$

$$x' = x + a_x$$
$$R = R_1 + R_2 - 0.1 * |a_x|$$

$$R_1 = \begin{cases} (tp_1') \wedge (\neg tp_1) \wedge (x > 50) & : 40 - 0.2 * (x - 50) \\ (tp_1') \wedge (\neg tp_1) \wedge (x < 50) & : 40 - 0.2 * (50 - x) \\ (tp_1') \wedge (tp_1) & : 1.1 \\ else & : -2 \end{cases}$$

$$R_2 = \begin{cases} (tp_2') \wedge (\neg tp_2) \wedge (x > -50) & : 60 - 0.2 * (-x + 50) \\ (tp_2') \wedge (\neg tp_2) \wedge (x < -50) & : 60 - 0.2 * (x + 50) \\ (tp_2') \wedge (tp_2) & : 1.2 \\ else & : -1 \end{cases}$$

In Figure 5, we plot different value functions obtained by compressing with different levels — we note that in general larger ϵ results in a looser fit, but there are exceptions, owing to the greedy nature of successive pairwise merging for XADDs described in Section 3.

Mars Rover2D: In this multivariate version of a Mars Rover domain the rover is expected to follow a path. The position is represented by a pair of continuous variables (x, y). There is only one action, $move(a_x, a_y)$, where $|a_x| < 10$ and $|a_y| < 10$. The new position is given by $(x', y') = (x + a_x, y + a_y)$. The reward increases with x and decreases with the absolute value of y, that is:

$$R = \begin{cases} (x > y + 25) \wedge (x > -y + 25) \wedge (y > 0) : & -10 + x - y \\ (x > y + 25) \wedge (x > -y + 25) \wedge (y < 0) : & -10 + x + y \\ else : & -1 \end{cases}$$

In Figure 6, we can clearly see the effect of compression. In the 3D plots, a much simpler surface is obtained for the 5% error compression, and correspondingly, in the diagrams, the number of nodes is greatly reduced, which enables a much faster computation of XADD operations and the bounded error solution.

Inventory Control: In an inventory problem [15], we assume n continuous resources that can be bought and sold. There are n *order-i* actions for each resource, $1 \leq i \leq n$. The maximum amount of each resource that is sold on one iteration depends on a stochastic demand variable d that is true with 60% probability. The reward is equal to the sum of the resources sold in this iteration.The resource x_i' for action *order-i* is given by:

$$x_i' = \begin{cases} (d') \wedge (x_i > 150) & : x_i + 200 - 150 \\ (d') \wedge (x_i < 150) & : 200 \\ (\neg d') \wedge (x_i > 50) & : x_i + 200 - 50 \\ (\neg d') \wedge (x_i < 50) & : 200 \end{cases}$$

and for other resources x_j', $1 \leq j \leq n$, $j \neq i$:

$$x_j' = \begin{cases} (d') \wedge (x_j > 150) & : x_j - 150 \\ (d') \wedge (x_j < 150) & : 0 \\ (\neg d') \wedge (x_j > 50) & : x_j - 50 \\ (\neg d') \wedge (x_j < 50) & : 0 \end{cases}$$

Figure 7 shows the time, space, and actual error of the BASDP solutions vs. the exact solution for one Mars Rover2D domain and one Inventory Control domain. In the space plots (left), we note how the approximation compresses the XADD significantly, even for small ϵ. We witness approximately 10× savings in time over the exact solution even for small ϵ and when we examine the actual error (right) of the BASDP solutions (compared to the exact solution), we see that it tends to be less than $\frac{1}{3}$ of the BASDP error bound.

6 Related Work

Boyan and Littman [4] presented the first exact solution for 1D continuous HMDPs with discrete actions, linear reward and piecewise dynamics while Feng *et al* [7] generalized this solution for a subset of multivariate HMDPs where all piecewise functions had to have rectilinear piece boundaries (i.e., general linear inequalities like $x + y > 0$ where disallowed) and actions were discrete. Li and Littman [10] extended Feng *et al*'s model to the case of bounded approximation using rectilinear piecewise *constant* functions that could not produce the low-error linear approximations in Figures 1(a,b) or 3. In addition, all of these methods could only provide (approximately) optimal solutions for a rectilinear subset of *discrete* action HMDPs in comparison to our more general setting of *continuous* action HMDPs with linear piecewise dynamics and rewards building on the work of Zamani *et al* [17].

An alternative bounded error HMDP solution is the phase-type approximation of Marecki *et al* [11] which can arbitrarily approximate 1D continuous MDP solutions but which does not extend to multivariate settings or continuous actions. In an approximate linear programming approach using basis functions, Kveton *et al* [9; 8] explore bounded approximations and learnable basis functions for HMDPs but cannot provide *a priori* guarantees on the maximum allowed error in a solution as we can in our BASDP framework. Munos and Moore [13] take a variable resolution discretization approach to refining value functions and policies, but these methods are based on rectilinear partitioned kd-trees which can consume prohibitive amounts of space to approximate the simple oblique piecewise linear function of Figure 2, represented exactly as a four node XADD.

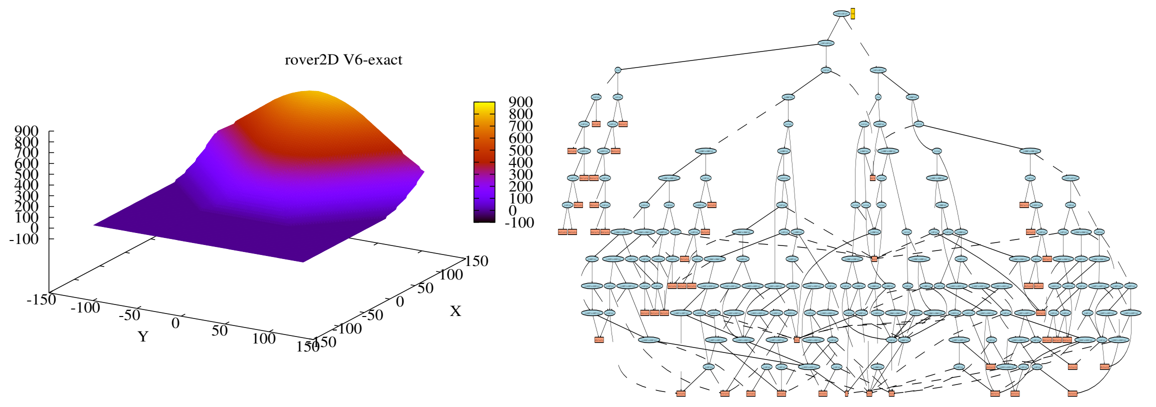
In an orthogonal direction to the work above, Meuleau *et al* [12] investigate a search-based dynamic programming solution to HMDPs that is restricted to optimality over a subset of initial states. We note that this approach admits *any* dynamic programming backup and value representation and hence can be combined with BASDP and XADDs as proposed here — an interesting avenue for future work.

7 Concluding Remarks

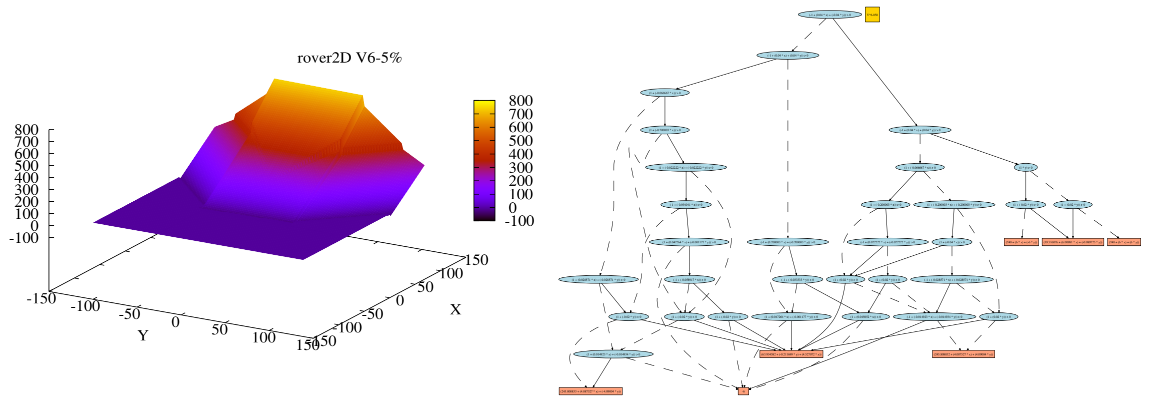
In this work, we introduced a novel bounded approximate symbolic dynamic programming (BASDP) algorithm for HMDPs based on XADD approximation, where we contributed a bounded error compression technique for XADDs involving the solution of a constrained bilinear saddle point problem. After exploiting a number of key insights in the structure of this problem, we were able to show that it could be solved to optimality in finite time via constraint generation in a linear programming framework (despite having an apparent infinite set of potential constraints). Empirically, this BASDP solution yielded order-of-magnitude speedups over the exact solution in exchange for a small approximation error, thus vastly expanding the range of HMDPs for which bounded error approximate solutions are possible.

Acknowledgments

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program. This work was supported by the Brazillian agency FAPESP (grant 2012/10861-0).



(a) Value at 6th iteration for exact SDP.



(b) Value at 6th iteration for 5% approximate SDP.

Figure 6: Value function at iteration 6 for the MARS ROVER2Ddomain; *(a)* Exact value function; *(b)* Approximate value function with error bounded 5% per iteration; *(left)* 3D Plots; *(right)* XADD Diagrams.

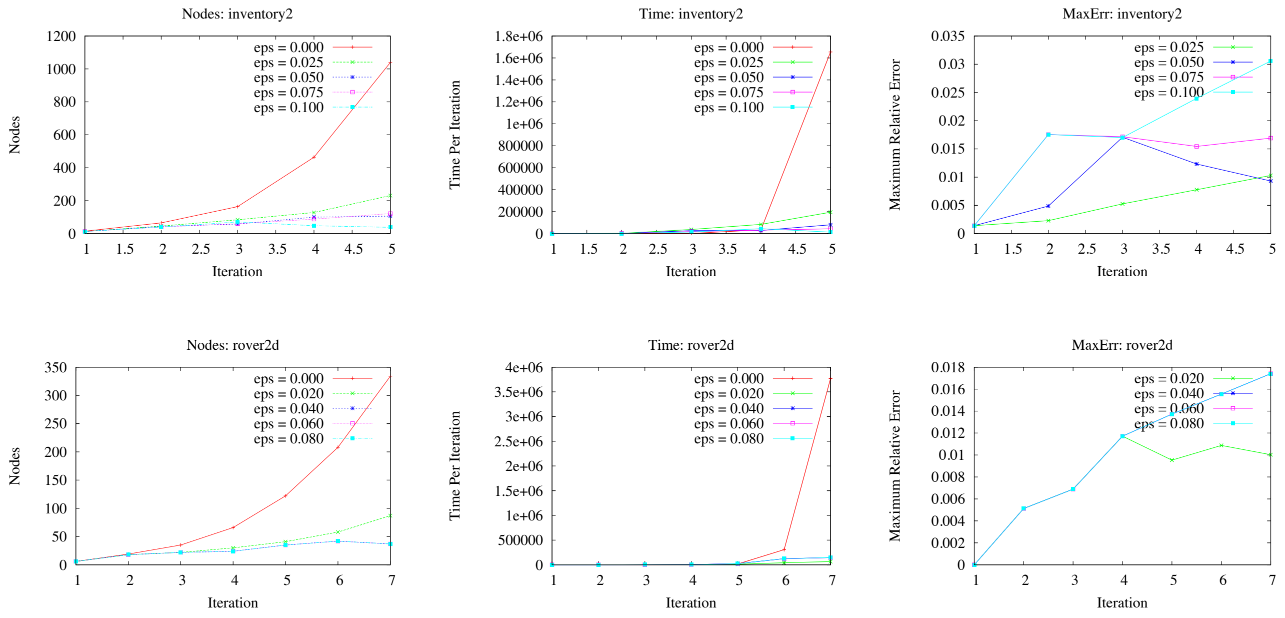


Figure 7: Performance plots for MARS ROVER2D and INVENTORY CONTROL2 with 5 different relative errors (eps): *(left)* Space (number of Nodes); *(middle)* Time (miliseconds); *(right)* Maximal error as fraction of the max value.

References

[1] R. Iris Bahar, Erica Frohm, Charles Gaona, Gary Hachtel, Enrico Macii, Abelardo Pardo, and Fabio Somenzi. Algebraic Decision Diagrams and their applications. In *IEEE /ACM International Conference on CAD*, 1993.

[2] Richard E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

[3] Craig Boutilier, Thomas Dean, and Steve Hanks. Decision-theoretic planning: Structural assumptions and computational leverage. *JAIR*, 11:1–94, 1999.

[4] Justin Boyan and Michael Littman. Exact solutions to time-dependent MDPs. In *Advances in Neural Information Processing Systems NIPS-00*, pages 1026–1032, 2001.

[5] John L. Bresina, Richard Dearden, Nicolas Meuleau, Sailesh Ramkrishnan, David E. Smith, and Richard Washington. Planning under continuous time and resource uncertainty:a challenge for ai. In *Uncertainty in Artificial Intelligence (UAI-02)*, 2002.

[6] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

[7] Zhengzhu Feng, Richard Dearden, Nicolas Meuleau, and Richard Washington. Dynamic programming for structured continuous markov decision problems. In *Uncertainty in Artificial Intelligence (UAI-04)*, pages 154–161, 2004.

[8] Branislav Kveton and Milos Hauskrecht. Learning basis functions in hybrid domains. In *In Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*, pages 1161–1166, Boston, USA, 2006.

[9] Branislav Kveton, Milos Hauskrecht, and Carlos Guestrin. Solving factored mdps with hybrid state and action variables. *Journal Artificial Intelligence Research (JAIR)*, 27:153–201, 2006.

[10] Lihong Li and Michael L. Littman. Lazy approximation for solving continuous finite-horizon mdps. In *National Conference on Artificial Intelligence AAAI-05*, pages 1175–1180, 2005.

[11] Janusz Marecki, Sven Koenig, and Milind Tambe. A fast analytical algorithm for solving markov decision processes with real-valued resources. In *International Conference on Uncertainty in Artificial Intelligence IJCAI*, pages 2536–2541, 2007.

[12] Nicolas Meuleau, Emmanuel Benazera, Ronen I. Brafman, Eric A. Hansen, and Mausam. A heuristic search approach to planning with continuous resources in stochastic domains. *Journal Artificial Intelligence Research (JAIR)*, 34:27–59, 2009.

[13] Andrew Moore Remi Munos. Variable resolution discretization in optimal control. *Machine Learning*, 49, 2–3:291–323, 2002.

[14] Scott Sanner, Karina Valdivia Delgado, and Leliane Nunes de Barros. Symbolic dynamic programming for discrete and continuous state mdps. In *Proceedings of the 27th Conference on Uncertainty in AI (UAI-2011)*, Barcelona, 2011.

[15] Herbert E Scarf. Inventory theory. *Operations Research*, 50(1):186–191, 2002.

[16] Robert St-Aubin, Jesse Hoey, and Craig Boutilier. APRICODD: Approximate policy construction using decision diagrams. In *NIPS-2000*, pages 1089–1095, Denver, 2000.

[17] Zahra Zamani, Scott Sanner, and Cheng Fang. Symbolic dynamic programming for continuous state and action mdps. In Jörg Hoffmann and Bart Selman, editors, *AAAI*. AAAI Press, 2012.

On MAP inference by MWSS on Perfect Graphs

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

Finding the most likely (MAP) configuration of a Markov random field (MRF) is NP-hard in general. A promising, recent technique is to reduce the problem to finding a maximum weight stable set (MWSS) on a derived weighted graph, which if perfect, allows inference in polynomial time. We derive new results for this approach, including a general decomposition theorem for MRFs of any order and number of labels, extensions of results for binary pairwise models with submodular cost functions to higher order, and an exact characterization of which binary pairwise MRFs can be efficiently solved with this method. This defines the power of the approach on this class of models, improves our toolbox and expands the range of tractable models.

1 INTRODUCTION

Markov random fields (MRFs), also termed undirected probabilistic graphical models, are a central tool in machine learning with wide use in many areas including speech recognition, vision and computational biology. A model (V, Ψ) is specified by a set of n variables $V = \{X_1, \dots, X_n\}$ together with (log) potential functions over subsets c of V, $\Psi = \{\psi_c : c \in C \subseteq \mathcal{P}(V)\}$, where $\mathcal{P}(V)$ is the powerset of V. In this paper, each variable X_i may take finite k_i possible values which we label $\{0, \dots, k_i - 1\}$. Write $x = (x_1, \dots, x_n)$ for one particular complete configuration and x_c for a configuration just of the variables in c. A potential function ψ_c maps each possible setting x_c of its variables c to a real number $\psi_c(x_c)$.

Identifying a configuration of variables that is most likely, termed *maximum a posteriori* or MAP inference, is very useful in many contexts, yet in general is NP-hard (Shimony, 1994). In our notation this is the combinatorial problem of identifying[1]

$$x^* = \operatorname*{arg\,max}_{x=(x_1,\dots,x_n)} \sum_{c \in C} \psi_c(x_c). \qquad (1)$$

In general, an MRF may be considered a hypergraph together with associated ψ_c functions (see section 2.1 for definitions). A popular alternative representation is a factor graph, which is a bipartite graph where the variables V form one stable partition and each $c \in C$ is a node in the other partition, with an edge from c to each variable it contains. In the special case that all variables X_i take values only in $\mathbb{B} = \{0, 1\}$, the model is said to be *binary*. If $|c| \le 2 \ \forall c \in C$ then the model is *pairwise*. Binary pairwise models play a key role in computer vision both directly and as critical subroutines in solving more complex problems (Pletscher & Kohli, 2012). Note that it is possible to convert a general MRF into an equivalent binary pairwise model (Yedidia et al., 2001; Ravikumar & Lafferty, 2006), though this may lead to a much larger state space.

1.1 RELATED WORK

It is well-known that the MAP estimate can be recovered for junction trees and acyclic graphical models using dynamic programming, junction tree algorithms, as well as max-product message passing (Bertelé & Brioschi, 1972; Pearl, 1988; Wainwright & Jordan, 2008). Such approaches hinge on the graph having bounded dimension or low tree-width, which is indeed the case for many useful Bayesian networks. Subsequently, graphical models with more general (and often dense) topologies yet whose potentials are constrained to be binary pairwise associative (ferromagnetic) functions were shown to be solvable efficiently using graph-

[1]This formulation assumes each configuration has probability > 0. When this is not the case, typically 0 may be replaced by a sufficiently small ϵ. Also *cost* functions are the negative of our ψs, thus submodular cost functions are equivalent to supermodular ψs.

cuts or network flow (Greig et al., 1989; Boykov & Kolmogorov, 2004). Many computer vision and image processing problems can be handled by this class of models. More recently, MAP estimation for cyclic graphical models involving matching and b-matching problems[2] was shown to be solvable efficiently using the max-product algorithm (Bayati et al., 2005; Huang & Jebara, 2007; Sanghavi et al., 2008; Bayati et al., 2008). In previous work, these three known cases were all shown to compile to a maximum weight stable set problem on a perfect graph, which is known to be solvable in polynomial time (Jebara, 2009; Jebara, 2012). This paper derives new results for this approach, first described in (Jebara, 2009; Sanghavi et al., 2009), and examines which other models may be handled in this manner.

An earlier method examining triangulated[3] microstructure graphs was presented (Jégou, 1993) in the context of constraint satisfaction problems (CSPs). Valued CSPs (VCSPs) use soft constraints with explicit costs, and are closely related to MAP inference problems. Many other techniques have been developed, including optimal soft arc consistency (Cooper et al., 2010), belief propagation (Weiss et al., 2007) and linear program relaxations (Sontag et al., 2008), which may be considered to proceed through identifying helpful reparameterizations (see section 2.5).

1.2 CONTRIBUTION AND SUMMARY

In section 2, we present important preliminary terms and results from graph theory, and on the approach of MAP inference via MWSS on a derived graph called an NMRF (a *nand Markov random field*, see section 2.4). This reviews previous work and introduces some novel concepts needed later.

In section 3 we derive a general decomposition theorem for mapping MRFs to NMRFs, which can be used to break apart a complex problem into smaller parts that overlap only on single variables. In situations where there are only a few of these overlapping variables, one could solve each subproblem and use a brute force enumeration approach over all combinations of the overlapping variables to find the global optimum, but this is clearly exponential in the number of overlapping variables. Our approach, in contrast, runs in polynomial time even for $\Omega(n)$ overlapping variables. This general result applies for potential functions ψ_c of any order, and variables with any number of labels. Note that each subproblem could have high treewidth.

[2]These graphical models involve topological constraints as well as various constraints on the potential functions (not simply associativity or submodularity).

[3]Triangulated, or chordless, graphs are a subclass of perfect graphs.

In section 4 we apply this general result specifically to pairwise models, focusing on the binary case to derive features of corresponding NMRFs. Applying these results, we proceed in section 5 to build towards Theorem 19, which provides a precise characterization of which binary pairwise MRFs map to perfect NMRFs for all valid ψ_c, and hence are amenable to this approach for efficient MAP inference.

In section 6 we explore a different direction, generalizing the previous result (Jebara, 2012) that binary pairwise MRFs with submodular cost functions can always be mapped to bipartite NMRFs (a special case of perfect graphs, admitting faster inference). We show that a bipartite pruned NMRF is obtained, for any topology, for third order cost functions iff they are submodular, and demonstrate that for interactions of order ≥ 4, submodularity is a necessary but strictly insufficient condition. Section 7 provides a conclusion and outlines future work.

2 BACKGROUND

2.1 TERMS FROM GRAPH THEORY

We follow standard definitions and omit some familiar terms, see (Diestel, 2010). A *graph* $G(V, E)$ is a set of vertices V, and edges $E \subseteq V \times V$. Let $n = |V|$ and $m = |E|$. Throughout this paper, unless otherwise specified, all graphs are finite and *simple*, that is a vertex may not be adjacent to itself (no loops) and each pair of vertices may have at most one edge (no multiple edges).

The *complete* graph on n vertices, written K_n, has all $\binom{n}{2}$ edges. A *path* of length n is a graph P_n with n edges connecting $n+1$ vertices as $v_1 - v_2 - \cdots - v_n - v_{n+1}$. An *induced subgraph* $H(U, F)$ of a graph $G(V, E)$ is a graph on some subset of the vertices $U \subseteq V$, inheriting all edges with both ends in U, so $F = \{(v, w) \in E : v, w \in U\}$. The union of two subgraphs, $H_1(V_1, E_1)$ and $H_2(V_2, E_2)$ of a graph $G(V, E)$, written $H_1 + H_2$, is the induced subgraph of G on $V_1 \cup V_2$.

A *hypergraph* (V, E) is a generalization of a graph where the elements of E are any non-empty subsets of V, not necessarily of size two. A general MRF may be regarded as a hypergraph (V, C) together with functions $\{\psi_c\}\ \forall c \in C$. For the special case of a pairwise model, the structural relationships are naturally interpreted as a graph.

A graph is *connected* if there is a path connecting any two vertices. A *cut vertex* of a connected graph G is a vertex $v \in V$ such that deleting v disconnects G. A graph is *2-connected*, equivalently *biconnected*, if it is connected and contains no cut vertex. A *block* is a maximal connected subgraph with no cut vertex of the subgraph. Every block is either K_2 (two vertices joined by an edge) or a maximal 2-connected subgraph containing a cycle. Different blocks of G overlap on at most one vertex, which must be a cut vertex. Hence G can be written as the union of its blocks with every edge in exactly one block. These blocks are connected without cycles in the *block tree* for G.

A *stable* set in a graph is a set of vertices, no two of which are adjacent. A *weighted graph* (V, E, w) is a graph with a nonnegative real value for each vertex, called its *weight* $w(v)$. A *maximum weight stable set (MWSS)* is a stable set with maximum possible weight. A *maximal maximum weight stable set (MMWSS)* is a MWSS of maximal cardinality (this is useful in our context since, after reparameterization, we may have many nodes with 0 weight, see sections 2.4 and 2.5).

A *clique* in a graph is a set of vertices, of which every pair is adjacent. The *clique number* of a graph G, written $\omega(G)$, is the maximum size of a clique in G.

The *complement* of a graph $G(V, E)$ is the graph $\bar{G}(V, F)$ on the same vertices with an edge in F iff it is not in E. Hence a clique is the complement of a stable set and vice versa.

A *coloring* of a graph is a map from its vertices to the integers (considered the colors of the vertices) such that no two adjacent vertices share the same color. The *chromatic number* of a graph G, written $\chi(G)$, is the minimum number of colors required to color it. Observe that clearly $\chi(G) \geq \omega(G)$ for any graph G.

A graph G is *perfect* iff $\chi(H) = \omega(H)$ for all induced subgraphs H of G. As examples, any bipartite or chordal graph is perfect. Related concepts (see Theorem 5) are: a *hole* in a graph G is an induced subgraph which is a cycle of length ≥ 4 (note this means the cycle must be chordless); an *antihole* is an induced subgraph whose complement is a hole. A hole or antihole is *odd* if it has an odd number of vertices. Note that, as a special case, a hole with 5 vertices is equivalent to an antihole of the same size. It is easily shown that odd holes and antiholes are not perfect.

2.2 FURTHER TERMS

This section may be skipped on a first reading, and referred to later for definitions.

A *clique group* for a set of variables c is a clique in an NMRF corresponding to all possible settings x_c of those variables of its MRF, see section 2.4.

An *snode* is a node in an NMRF relating to a setting of a single variable from its MRF. Equivalently, it is a node from a clique group deriving from $c = \{X_i\}$ for some i. An *enode* is a node from a clique group deriving from some $c \in C$ with $|c| \geq 2$. For example, when considering binary pairwise models, an enode derives from an edge of the MRF.

For a graph (V, E), if $X \subseteq V$ and $v \in V \setminus X$ then v is *complete* to X if v is adjacent to every member of X. If $X, Y \subseteq V$ are disjoint, then X is *complete* to Y if every vertex in X is complete to Y.

A *cutset* S of a graph G is a set of vertices $S \subseteq V(G)$ s.t. $G \setminus S$ is disconnected. A *star-cutset* S of G is a cutset s.t. $\exists$ some $x \in S$ s.t. x is complete to $S \setminus \{x\}$.

A *signed graph* (Harary, 1953) is a graph (V, E) together with one of two possible signs for each edge. This is a natural structure when considering binary pairwise models, where we characterize edges as either *associative* or *repulsive*, see section 2.6. When discussing signed graphs, we use the notation $\oplus$ to show an associative edge, and $\ominus$ for a repulsive edge. For example, $x \oplus y \ominus z$ is a graph with 3 vertices x, y and z, and two edges, where x and y are adjacent via an associative edge, and y and z are adjacent via a repulsive edge.

A *frustrated cycle* in a signed graph is a cycle with an odd number of repulsive edges.

A B_R structure (see Figure 1 for an example) is a signed graph over variables V with associative edges E_A and repulsive edges E_R s.t. (V, E_R) is bipartite and $\exists$ a disjoint bipartition $V = V_1 \cup V_2$ with all E_R crossing between the partitions $V_1 - V_2$, and no E_A crossing between them. Either E_A or E_R may be empty, so for example, this includes any signed graph with only associative edges.

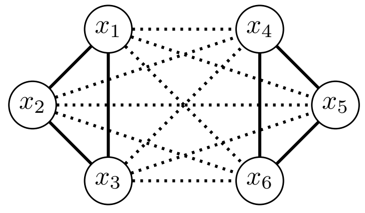


Figure 1: A B_R structure. Solid (dashed) edges are associative (repulsive). Deleting any edges maintains the B_R property.

A $T_{m,n}$ structure (see Figure 2 for an example) is a 2-connected signed graph containing $m + n \geq 1$ triangles on a common base given by: 2 base vertices s, t connected via a repulsive edge, so $s \ominus t$; together with $m \geq 0$ vertices r_i, each adjacent only to s and t via repulsive edges, so $s \ominus r_i \ominus t$; and $n \geq 0$ vertices a_i, each adjacent only to s and t via associative edges, so $s \oplus a_i \oplus t$. Note $T_{m,n}$ would be bipartite, with $\{s, t\}$ as one partition and all other vertices in the other, except that we have the repulsive edge $s \ominus t$.

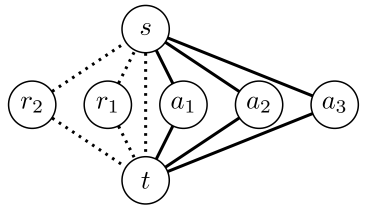


Figure 2: A $T_{m,n}$ structure with $m = 2$ and $n = 3$. Solid (dashed) edges are associative (repulsive).

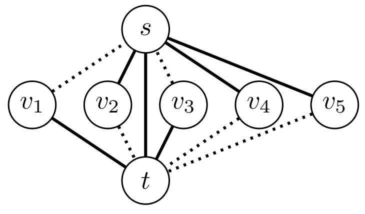


Figure 3: A U_n structure with $n = 5$. Solid (dashed) edges are associative (repulsive).

A U_n structure (see Figure 3 for an example) is a 2-connected signed graph containing $n \geq 1$ triangles on a common base given by: 2 base vertices s, t connected via an associative edge, so $s \oplus t$; together with $n \geq 1$ vertices v_i, each adjacent only to s and t via one associative and one repulsive edge (either way), so either $s \oplus v_i \ominus t$ or $s \ominus v_i \oplus t$.

Note that U_1 is the same as $T_{0,1}$ but this is the only overlap. In Lemma 18, we show that $T_{m,n}$ and U_n structures are the only 2-connected signed graphs containing a frustrated cycle that map to a perfect NMRF.

2.3 PROPERTIES OF PERFECT GRAPHS

2.3.1 Complexity of MWSS

Our approach to MAP inference is to reduce the problem to finding a maximum weight stable set on a derived weighted graph, as described in section 2.4. This is helpful only if we can find a MWSS efficiently, yet in general this is still an NP-hard problem for a graph with N vertices. However, if the derived graph is perfect[4], then a MWSS may be found in polynomial time via the ellipsoid method (Grötschel et al., 1984).

Faster exact methods (Yildirim & Fan-Orzechowski, 2006) based on semidefinite programming are possible in $O(N^6)$ and are improved using primal-dual methods (Chan et al., 2009). Alternatively, linear programming can solve MWSS problems but requires $O(N^3\sqrt{n_K})$ time where n_K is the number of maximal cliques in the graph (Jebara, 2009; Jebara, 2012). Clearly, whenever n_K is small, linear programming can be more efficient than semidefinite programming. However, in the worst case, n_K may be exponentially large in N which makes linear programming useful only in some cases. Message-passing methods can also be applied for finding the maximum weight stable set in a perfect graph though they too become inefficient for graphs with many cliques (Foulds et al., 2011; Jebara, 2012).

[4]There are a few other classes of graphs that also admit efficient MWSS, such as *claw-free* graphs, where significant recent advances have been made (Faenza et al., 2011), but so far these have not been useful in analyzing MRFs.

Where other methods exist for solving exact MAP inference, the reduction to MWSS is typically not the fastest method, yet there is hope for improvement since the field is advancing rapidly, with significant breakthroughs in recent years (Chudnovsky et al., 2006; Faenza et al., 2011).

2.3.2 Other properties

There is a rich literature on perfect graphs. We highlight key results used later in this paper.

Theorem 1 ((Gallai, 1962))**.** *The graph obtained by pasting two perfect graphs on a clique is perfect.*

Theorem 2 ((Chvátal, 1985))**.** *The graph obtained by pasting two perfect graphs on a star-cutset is perfect.*

Theorem 3 (Substitution Lemma, (Lovász, 1972))**.** *The graph obtained by substituting one perfect graph for a vertex of another perfect graph is also perfect.*

Here, substituting H for x in G means deleting x and joining every vertex of H to those vertices of G which were adjacent to x.

Theorem 4 (Weak Perfect Graph Theorem, (Lovász, 1972))**.** *A graph is perfect iff its complement is perfect.*

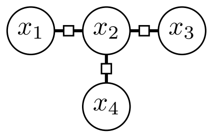
Theorem 5 (Strong Perfect Graph Theorem 'SPGT' (Chudnovsky et al., 2006))**.** *A graph is perfect iff it contains no odd hole or antihole.*

2.4 MAP REDUCTION TO MWSS

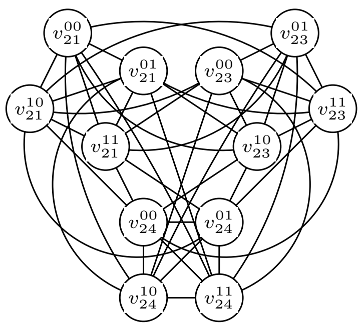
Given an MRF model (V, Ψ), construct a *nand Markov random field (NMRF)*, see (Jebara, 2009):

- A weighted graph $N(V_N, E_N, w)$ with vertices V_N, edges E_N and a weight function $w : V_N \to \mathbb{R}_{\geq 0}$.
- Each $c \in C$ of the original model maps to a *clique group* of N which contains one node for each possible configuration x_c, all pairwise adjacent.
- Generally, nodes are adjacent iff they have inconsistent settings for any variable X_i.
- Nonnegative weights of each node in N are set as $\psi_c(x_c) - \min_{x_c} \psi_c(x_c)$, see section 2.5 for an explanation of the subtraction.

See Figure 4 for an example. (Jebara, 2012) proved that a maximal cardinality set of consistent configuration nodes in N with greatest total weight, i.e. a MMWSS of N (see section 2.1), will identify a globally



(a) Input MRF



(b) Derived NMRF

Figure 4: An example of mapping an MRF with binary variables (shown as a factor graph) to an NMRF (subscripts denote the factor variables c and superscripts denote the configuration x_c).

consistent configuration of all variables of the original MRF that solves the MAP inference problem (1).

Sketch proof: (Slightly different to (Jebara, 2012), this will allow us to extend the result after discussing pruning in section 2.5.) A MMWSS S is consistent by construction and clearly contains at most one node from each clique group. It remains to show it has at least one from each clique group. Suppose a clique group has no representative. Identify a member of this group which could be added to S, establishing a contradiction since S is maximal, as follows: the group overlaps with some variables of S, copy the settings of these; for all other variables in the group, pick any setting. Note that if we do not insist on a maximal MWSS, it is possible that we do not get a representative for some clique groups and hence do not obtain a complete MAP configuration for the initial MRF.

2.5 REPARAMETERIZATIONS AND PRUNING

A *reparameterization* is a transformation

$$\{\psi_c\} \to \{\psi'_c\} \text{ s.t. } \forall x, \sum_{c\in C}\psi_c(x_c) = \sum_{c\in C}\psi'_c(x_c) + \text{constant}.$$

This clearly does not modify (1) but can be helpful to simplify the problem.

One particular reparameterization is to add a constant just to any ψ_c function, since any consistent configuration has exactly one setting for each group of variables c. Hence we may subtract the minimum $\psi_c(x_c)$ and assume that in each clique group of N, the minimum weight of a node is exactly zero. The earlier reduction result in section 2.4 holds provided we insist on a *maximal* MWSS. To find a MMWSS, it is sufficient first to remove or *prune* the zero weight nodes, find a MWSS on the remaining graph, then reintroduce a maximal number of the zero weight nodes while maintaining stability of the set. Different reparameterizations will yield different pruned NMRFs. By the earlier argument: MWSS will find one member from each of some of the clique groups, then we can always find one of the zero weight nodes to add from each of the remaining groups using any greedy method. Hence we have shown the following result.

Lemma 6. *MAP inference on an MRF is tractable provided $\exists$ an efficient reparameterization s.t. the MRF maps to a perfect pruned NMRF.*

2.6 SINGLETON TRANSFORMATIONS, BINARY PAIRWISE MRFS AND ASSOCIATIVITY

Another useful reparameterization is what we term a *singleton transformation*, which is a change in one or more ψ functions for a single variable, with corresponding changes to a higher order term which brings it to a convenient form.

Considering binary pairwise models only, it is easily shown that a reparameterization of an edge via singleton transformations, $\begin{pmatrix}\psi_{00} & \psi_{01}\\ \psi_{10} & \psi_{11}\end{pmatrix} \to \begin{pmatrix}\psi'_{00} & \psi'_{01}\\ \psi'_{10} & \psi'_{11}\end{pmatrix}$ is valid iff $\psi_{00}+\psi_{11}-\psi_{01}-\psi_{10} = \psi'_{00}+\psi'_{11}-\psi'_{01}-\psi'_{10}$. Hence this quantity, which we call the *associativity* of the edge, is invariant with respect to any singleton transformation, and thus is well defined.

We describe an edge as either *associative*[5], in which case it tends to pull its two end vertices toward the same value, or *repulsive*, in which case it tends to push its two end vertices apart to different values, according to whether its associativity is > 0 or < 0. An edge with 0 associativity may be removed since we may transform its edge potential to the zero matrix. A binary pairwise model is associative iff every one of its edges is associative.

An associative edge may be reparameterized s.t. three of its entries are 0, and therefore may be pruned, leaving only either ψ'_{00} or ψ'_{11} (or both, though for our purposes of mapping to a perfect NMRF, it is always easier to prune more nodes) with a positive value. Similarly, we may reparameterize a repulsive edge $x \ominus y$ to leave only a $(x=0, y=1)$ or $(x=1, y=0)$ node.[6]

[5]Other equivalent terms used are *attractive*, *ferromagnetic* or *regular*. This is equivalent to ψ for the edge being supermodular, or having submodular cost function.

[6]For repulsive edges, selecting one or other form is exactly analogous to choosing an *orientation* of the edge, $x \to y$ or $x \leftarrow y$. Further, such enodes from repulsive edges are adjacent iff their directed edges connect 'head to tail', hence the induced subgraph of an NMRF on these

2.7 SINGLETON CLIQUE GROUPS

Since typically we would like to allow any finite values for singleton potential functions, and singleton transformations as described in section 2.6 without restriction, in this paper we assume that any NMRF includes the complete clique group for each of the single variables of its MRF. In particular contexts, however, one may drop this requirement, and since this would remove nodes from the NMRF, it can only help to show perfection (any induced subgraph of a perfect graph is perfect), though then sometimes care must be taken to confirm the decomposition result of Theorem 7.

3 NEW RESULTS FOR ALL MRFS

Theorem 7 (MRF Decomposition)**.** *If $MRF_A(V_A, \Psi_A)$ and $MRF_B(V_B, \Psi_B)$ both map to perfect NMRFs N_A and N_B, and have exactly one variable s in common, i.e. $V_A \cap V_B = \{s\}$, with consistent ψ_s, then the combined $MRF'(V_A \cup V_B, \Psi_A \cup \Psi_B)$ maps to an NMRF N' which is also perfect. The converse is true by the definition of perfect graphs.*

Proof. [7] See section 2.2 for notation. We may assume both Ψ_A and Ψ_B contain the same ψ_s forming the complete s clique group K_s in N_A and N_B (see section 2.7, though in fact this Theorem holds more generally, provided only that both N_A and N_B have the same nodes from the clique group for s).

Let the possible values of s be $\{0, \dots, k-1\}$, and s_i be the snode corresponding to $(s = i)$. Let A_i be all those vertices of $N_A \setminus \{s_i\}$ which have setting $s = i$, similarly define B_i for N_B. Observe that A_i is complete to A_j for all $i \neq j$, and similarly for B_i. N' is the result of pasting N_A and N_B on K_s, together with all edges from A_i to B_j if $i \neq j$.

Hence N' admits a star-cutset given by $X = K_s + A_1 + \dots + A_k + B_1 + \dots + B_k$ with s_0 complete to $X \setminus \{s_0\}$. Thus by Theorem 2, it is sufficient to show that $N_A + X$ and $N_B + X$ are each perfect. But this is true by Theorem 3, since $N_A + X = N_A + B_1 + \dots + B_k$ may be obtained from N_A by substituting (via Theorem 3) $B_i + s_i$ for s_i, $i = 1, \dots, k$; and similarly for N_B. □

3.1 BLOCK DECOMPOSITION

Theorem 7 is a powerful tool for analyzing MRFs of any order and number of labels. As a special case, we have an immediate corollary.

Theorem 8. *A pairwise MRF maps to a perfect NMRF for all valid ψ iff each of its blocks maps to a perfect NMRF.*

This provides an elegant way to derive a previous result (Jebara, 2009):

Theorem 9. *A pairwise MRF whose graph structure is a tree (i.e. no cycles) maps to a perfect NMRF.*

Proof. By Theorem 8, we need only consider one edge together with its two end vertices (then use induction). The edge clique group together with each one of the singleton clique groups is the complement of a bipartite graph, hence is perfect (by Theorem 4). Now paste the two together on the edge clique group to show the whole is perfect (by Theorem 1). □

We show the following further general result.

Lemma 10. *Neither an odd hole H nor an odd antihole A in a NMRF can contain ≥ 2 members, say s_1 and s_2, of any singleton clique group.*

Proof. In H, s_1 and s_2 must be next to each other, then moving out round H one node in each direction, we cannot avoid a chord, contradiction. In A, there must be at least 2 nodes between s_1 and s_2 in at least one direction. Taking this way round A, the node next to s_1 must be adjacent to s_2 but not s_1, so has setting $s = 1$. Continuing round A, the next node must be adjacent to s_1, so must have an s value $\neq 1$ but then it is adjacent to its predecessor, contradiction. □

4 NEW RESULTS FOR BINARY PAIRWISE MRFS

Lemma 11. *Let M be a binary pairwise MRF. $\exists$ a reparameterization s.t. M maps to perfect pruned NMRF $\Leftrightarrow$ $\exists$ a reparameterization with just one enode per edge in the pruned NMRF which is perfect.*

Proof. ($\Leftarrow$) is clear. ($\Rightarrow$) see section 2.6. With a standard reparameterization, we may always achieve just one pruned enode (either 00 or 11 for associative, 01 or 10 for repulsive) from those already present. The result follows from the definition of a perfect graph. □

Therefore henceforth, when referring to a pruned NMRF of a binary pairwise MRF, we may assume just one enode per edge.

Lemma 12. *An antihole A of size ≥ 7 can never occur in a pruned NMRF N from a binary pairwise MRF M.*

Proof. Suppose A exists containing an snode, WLOG say s_0. This must be adjacent to ≥ 4 nodes in A, all

repulsive enodes is exactly a directed line graph of (V, E_R).

[7]This proof, due to Maria Chudnovsky, is shorter and neater than the authors' original.

of which must have $s = 1$ settings. The 2 closest to s_0 around A one way must both be adjacent to the closest to s_0 around A the other way, which cannot be achieved, hence A must contain only enodes. By Lemma 11, we have only one enode per edge of M. Two enodes are adjacent in N if they have one end in common with different settings - since only 2 settings are possible, a triangle in N must derive from edges in M that formed a triangle. Given 2 enodes which are adjacent, there is exactly one possible third enode with which they can form a triangle (e.g. for $s0t1$ and $t0u0$, $s1u1$ is the unique third possible enode). Yet A must contain ≥ 2 triangles which have the same 2 members but a different third member, contradiction. □

Since an antihole of size 5 is equivalent to a hole of the same size, SPGT (Theorem 5) gives the following.

Lemma 13. *For a binary pairwise MRF, a pruned NMRF is perfect ⇔ it contains no odd hole.*

5 WHICH BINARY PAIRWISE MRFS YIELD PERFECT NMRFS

By Theorem 8, we need only consider 2-connected graphs G (considering both associative and repulsive edges), and by Lemma 13 we need only check for odd holes. G either contains a frustrated cycle or does not. If it does, we shall see that G must have the form $T_{m,n}$ or U_n. If not, we show G must have the form B_R. See section 2.2 for definitions.

Lemma 14 ((Harary, 1953))**.** *The following are equivalent properties for a signed graph G on vertices V:*

1. *G contains no frustrated cycle*
2. *G is of the form B_R*
3. *G is flippable to fully associative*

(1)⇔(2) by a variant of the standard proof that a graph is bipartite iff it has no odd cycle, considering repulsive edges. (3) means ∃ some subset $S \subseteq V$ s.t. if we replace each $X_i \in S$ by $Y_i = 1 - X_i$, and modify potential functions accordingly, thereby flipping the nature of each edge incident to X_i between associative and repulsive, then all edges of G can be made associative. (2)⇔(3) by setting S as either partition.

Theorem 15. *A binary pairwise MRF with the form B_R maps efficiently to a bipartite NMRF N.*

Proof. Let the partitions of the variables be S and T with snodes $\{s_i^0, s_i^1\}$ from S, and $\{t_j^0, t_j^1\}$ from T. Choose a reparameterization s.t. any associative edge $x \oplus y$ maps to an enode $(x = 0, y = 0)$, and for any repulsive edge pick either form. Hence in N we have:

$\{e_i\}$ associative enodes from S, form $(s_i = 0, s_j = 0)$,
$\{f_i\}$ associative enodes from T, form $(t_i = 0, t_j = 0)$,
$\{a_i\}$ repulsive enodes $S \to T$, form $(s_i = 0, t_j = 1)$,
$\{b_i\}$ repulsive enodes $S \leftarrow T$, form $(s_i = 1, t_j = 0)$.

Observe N is bipartite with partitions $\{a_i, s_i^0, t_j^1, e_i\}$ and $\{b_i, s_i^1, t_j^0, f_i\}$. □

We now explore the case that G has a frustrated cycle.

Lemma 16. *Any cycle C in a binary pairwise MRF generates an induced (chordless) cycle H in its NMRF N with size at least as great, and with the same parity (odd/even number of vertices) as the number of repulsive edges (odd/even) in the MRF's cycle.*
In particular, if M contains any frustrated cycle with ≥ 4 edges, or with 3 edges requiring any snode to link the enodes in N, then this maps to an odd hole in N.

Proof. By Lemma 11, we may assume just one enode in N per edge in G. Form a cycle H in N using the enodes corresponding to the edges of C, together with connecting snodes as required (if two enodes meet at a variable and have the same setting, add an snode with the opposite setting). Clearly H is chordless and $|H| \geq |C|$.

Pick some enode e_1 and orientation around H. Consider the *end parity* of e_1, that is the setting for the next variable around H. For subsequent enodes, to maintain end parity requires an even (odd) total number of nodes , including possible snodes, for associative (repulsive) edges, and the reverse to flip end parity. Let a_m and a_f be the number of times end parity is maintained and flipped respectively using all associative edges around H, and similarly define r_m and r_f for all repulsive edges. In order to connect to the other end of e_1 after traversing H requires in total (including e_1) an odd number of flips, hence $a_f + r_f \equiv 1 \pmod 2$. The total number of nodes in H is comprised of the first enode together with all subsequent nodes, hence

$$\begin{aligned}|H| &\equiv 1 + 0.a_m + 1.a_f + 1.r_m + 0.r_f \pmod 2\\ &\equiv a_f + r_m + 1 \pmod 2 \equiv r_f + r_m \pmod 2.\end{aligned}$$ □

Using Lemmas 13 and 16 we show the following result.

Lemma 17. *Let M be a binary pairwise MRF that maps to an NMRF N. If N is not perfect then ∃ a frustrated cycle in M that maps to an odd hole in N. Hence, N is perfect ⇔ ∄ such a cycle in M.*

Proof. By Lemma 13, N contains an odd hole H. By Lemma 10, any snode in H is adjacent to two enodes, and hence H must have derived from a cycle in M. Lemma 16 completes the proof. □

Lemma 18. *The only 2-connected binary pairwise MRFs containing a frustrated cycle, that map to a perfect NMRF, are of the form $T_{m,n}$ or U_n.*

Proof. See section 2.2 for definitions. By Lemmas 16 and 17, we need only consider a frustrated triangle in M whose enodes in N require no connecting snodes. This triangle may have either (1) one repulsive and two associative edges, which we shall show must be of the form U_n or $T_{m,n}$ with $n \geq 1$, or (2) three repulsive edges, which we shall show must be of the form $T_{m,n}$.

It is simple to check that, in either case, a fourth vertex adjacent to all 3 vertices of the triangle, resulting in a K_4 clique, does not admit a reparameterization that avoids a frustrated cycle requiring connecting snodes.

Case 1: Triangle with one repulsive edge. We have a U_1 structure. Let the configuration in the MRF be $s \oplus t \ominus v_1 \oplus s$. In order to avoid connecting snodes in N, we must have one of the following two reparameterizations: $\{(s=0,t=0),(t=1,v_1=0),(v_1=1,s=1)\}$ or $\{(s=1,t=1),(t=0,v_1=1),(v_1=0,s=0)\}$. Once one edge has been selected, the others can follow in only one way. Consider what may be added to this graph while remaining 2-connected and avoiding a frustrated cycle with ≥ 4 edges. Any additional vertex v_2 must be attached by disjoint paths to at least 2 vertices x and y of the triangle. If either path has length ≥ 2 then, by choosing one or other path in the original U_1 from x to y, we always find a frustrated cycle with ≥ 4 edges, leading to an odd hole. Using the argument from the preceding paragraph, v_2 must be adjacent to exactly 2 vertices of U_1. If these vertices are connected by an associative edge, we now have U_2; otherwise we have $T_{0,2}$. Checking cases now shows that the only way to add further vertices results in U_n or $T_{m,n}$ structures, with any $m \geq 0, n \geq 1$ allowed.

Case 2: Triangle with three repulsive edges. We have $T_{1,0}$. Similar reasoning to case 1 shows that the only possibilities are $T_{m,n}$ for any $m \geq 1, n \geq 0$. □

Taking the results of this section together, we have the following characterization.

Theorem 19. *A binary pairwise MRF maps to a perfect NMRF for all valid ψ_c iff each of its blocks (using all edges) has the form B_R, $T_{m,n}$ or U_n.*

5.1 REMARKS

Theorem 19 certainly has theoretical value in establishing the boundaries of the MWSS approach for this class of MRFs. Further, it broadens the landscape of tractable models. Each of the three block categories is itself tractable by other methods *in isolation*: QPBO (Rother et al., 2007) is guaranteed to be able to handle a B_R structure (though not $T_{m,n}$ or U_n), or indeed a B_R structure may be flipped to yield a fully associative model which can be solved with any appropriate technique such as graph cuts; and each $T_{m,n}$ or U_n has low tree width so admits traditional inference methods. To our knowledge, however, our approach is the first to be able to handle an MRF containing $\Omega(n)$ of these structures, including high tree width B_R sections, automatically in polynomial time.

5.1.1 Efficient Detection

Detecting if a binary pairwise MRF with topology (V,E) satisfies our conditions may be performed in time $O(|E|)$: identifying block structure is an application of DFS, then each block type may be efficiently checked. The $T_{m,n}$ and U_n structures are straightforward. For B_R, first test if it is bipartite using just E_R (an application of BFS). Next check each component by E_R to see that no E_A cross partitions. Then stitch together partitions from different components (if more than one) using E_A. If any E_A cross partitions then it is easy to see $\exists$ a frustrated cycle with ≥ 4 edges which would lead to an odd hole in the NMRF.

6 HIGHER ORDER SUBMODULAR

As noted in the introduction, (Jebara, 2012) has shown that a fully associative binary pairwise model, which is equivalent to a model with supermodular pairwise ψ functions (submodular cost functions), can always be reparameterized so as to yield a bipartite pruned NMRF. Indeed, we have seen in section 2.6 that, for each associative edge $x \oplus y$, one may reparameterize and prune the edge clique group so as to leave only either form $(x=0,y=0)$ or $(x=1,y=1)$. Here we extend the analysis to consider higher order models, still focusing on submodular cost functions over binary variables. We shall show that for potentials over 3 variables, a bipartite pruned NMRF is obtained for any topology iff all cost functions are submodular. Further, we show that submodularity is a necessary but strictly insufficient condition to obtain a bipartite pruned NMRF for all orders higher than 3.

Considering other approaches, this is similar to the result of (Zivny et al., 2009) that all order 3 submodular functions over Boolean variables can be represented by order 2 submodular functions using auxiliary variables, but this is not always true when the order > 3. Also, (Kolmogorov & Zabih, 2004) showed that submodularity was necessary for a function to be graph-representable. However, (Arora et al., 2012) recently demonstrated a novel graph cuts method for submodular cost functions of any order[8] over binary variables. Still, our result usefully clarifies the boundaries of our approach if we restrict to bipartite NMRFs only, and there is hope yet that a broader class of models may

[8]The time is exponential in the order of the potentials.

map to the wider class of perfect NMRFs.

6.1 NOTATION

Let ψ be an order k potential function over k binary variables $X = \{X_1, \ldots, X_k\}$. Let one setting be $x = (x_1, \ldots, x_k)$. Let $x - ij$ be a setting for all variables other than X_i and X_j. Let $\psi_x = \psi(X = x)$. Define the supermodularity s of ψ wrt X_i, X_j on the projection given by $x-ij$, as $s^{ij}_{x-ij} = \psi(X_i = 0, X_j = 0)+\psi(X_i = 1, X_j = 1) - \psi(X_i = 1, X_j = 0) - \psi(X_i = 0, X_j = 1)$ where all other variables in $X \setminus \{X_i, X_j\}$ are held fixed at $x - ij$.

Define $\alpha_k = \sum_{\text{all } 2^k \text{ settings of } x}(-1)^{\#0\text{s in } x}\psi_x$. Observe that for $k = 2$, this is the supermodularity s term. For $k = 3$, this is the difference between s with (any) one variable set to 0 and that with the same variable set to 1. For $k = 4$, we have the sum of two s terms minus two others, etc.

$\forall Y \subseteq \mathcal{P}(X)$, let O_Y and I_Y be weighted indicator functions. The O functions are 0 unless all of Y are 0. The I functions are 0 unless all of Y are 1. Otherwise, O_Y and I_Y take values Z_Y and A_Y, respectively. $Y = b$ means fix variables Y at value b where $b \in \mathbb{B} = \{0, 1\}$.

In order to map to a bipartite pruned NMRF for any topology at order k, we must be able to represent every ψ_x as the sum of a constant term and nonnegative[9] O and I indicator functions over all subsets of X, which correspond exactly to the nodes in the pruned NMRF (which is then clearly bipartite with stable sets corresponding to the $\{O_Y\}$ and $\{I_Y\}$).

6.2 RESULTS

Theorem 20. *For $k \geq 2$, mapping to a bipartite pruned NMRF for any topology $\Rightarrow$ ψ is supermodular, equivalently every projection of ψ onto two variables is supermodular.*

Proof. Given the ψ_x representation from the previous paragraph, consider which A_Y, Z_Y terms survive when a general supermodularity term s^{ij}_{x-ij} is computed. For some Y, analyze A_Y terms (a similar result holds for Z_Y terms): Y will include either none, one or two of the variables $\{X_i, X_j\}$. Consider the cases: If none, then A_Y does not feature in the s^{ij}_{x-ij} computation. If one, then we get plus A_Y (from the $X_i = X_j = 1$ term) minus A_Y (from the appropriate other term), so they cancel. Finally, if two, then we simply get plus the A_Y term. Hence for every s^{ij}_{x-ij}, it must be equal to the sum of some A_{Y_i} and Z_{Y_j} terms, all of which are constrained to be ≥ 0. Hence all supermodularity terms are ≥ 0. $\square$

Further, for $k = 4$, it is easily checked that $\alpha_k = A_X + Z_X$, where we require $A_X, Z_X \geq 0$, yet it also equals $s^{ij}_{x-ij=00} + s^{ij}_{x-ij=11} - s^{ij}_{x-ij=01} - s^{ij}_{x-ij=10}$ (for any 2 variables X_i, X_j), which may be positive but equally may be negative.[10] Similarly for all $k > 4$, we are not able to represent all supermodular ψ functions.

Theorem 21. *For general interactions over $k = 3$ variables, ψ is supermodular $\Leftrightarrow$ we obtain a bipartite pruned NMRF for any topology.*

Proof. ($\Leftarrow$) follows from Theorem 20. ($\Rightarrow$) we provide a constructive proof:[11]

If $\alpha_k \geq 0$, use only O_Y for $|Y| \geq 2$. Set $Z_X = \alpha_k$. For $|Y| = 2$, set $Z_Y = s^Y_1$. For $|Y| = 1$, set $Z_Y = \psi(Y = 0, (X \setminus Y) = 1) - \psi_{111}$. Set constant to ψ_{111} to observe we match ψ_x values $\forall x$. Now reparameterize all singleton terms and prune as usual, see section 2.5.

If $\alpha_k \leq 0$, use only I_Y for $|Y| \geq 2$. Set $A_X = -\alpha_k$. For $|Y| = 2$, set $A_Y = s^Y_0$. For $|Y| = 1$, set $A_Y = \psi(Y = 1, (X \setminus Y) = 0) - \psi_{000}$. As before, set constant to ψ_{000} to check values, then reparameterize all singleton terms and prune, see section 2.5. $\square$

7 CONCLUSIONS

The MWSS approach to MAP inference is an exciting, recent approach, leveraging the rapid progress in combinatorics. Here we have derived new general tools (section 3), defined the scope of the approach in an important, broad setting (sections 4 and 5), where we were able to extend the range of known tractable models, and clarified the power of mapping to bipartite NMRFs (section 6).

Future areas to explore include non-bipartite perfect NMRFs for higher order potentials, and variables with a greater number of labels.

Acknowledgments

We thank Maria Chudnovsky for helpful discussions and for the shorter proof of Theorem 7 shown, and the anonymous reviewers for valuable suggestions. This material is based upon work supported by the National Science Foundation under Grant No. 1117631.

[9] It is critical that the functions be nonnegative in order that the corresponding nodes in the NMRF are the only ones not pruned.

[10] An example of supermodular ψ for $k = 4$ where $\alpha_k < 0$: $\psi(x_1, x_2, x_3, x_4) = 0$ except $\psi(0, 0, 0, 0) = 2, \psi(1, 0, 0, 0) = \psi(0, 1, 0, 0) = \psi(0, 0, 1, 0) = \psi(0, 0, 0, 1) = 1$.

[11] In fact, as shown, we need use only either exclusively O_Y or I_Y nodes for $|Y| \geq 2$, which may further improve efficiency.

References

Arora, C., Banerjee, S., Kalra, P., & Maheshwari, S. N. (2012). Generic cuts: An efficient algorithm for optimal inference in higher order MRF-MAP. *ECCV (5)* (pp. 17–30).

Bayati, M., Borgs, C., Chayes, J., & Zecchina, R. (2008). On the exactness of the cavity method for weighted b-matchings on arbitrary graphs and its relation to linear programs. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*, L06001 (10pp).

Bayati, M., Shah, D., & Sharma, M. (2005). Maximum weight matching via max-product belief propagation. *IEEE International Symposium on Information Theory.*

Bertelé, U., & Brioschi, F. (1972). *Nonserial dynamic programming*. Academic Press. ISBN 0-12-093450-7.

Boykov, Y., & Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*, 1124–1137.

Chan, T.-H. H., Chang, K., & Raman, R. (2009). An SDP primal-dual algorithm for approximating the Lovász-theta function. *IEEE International Symposium on Information Theory.*

Chudnovsky, M., Robertson, N., Seymour, P., & Thomas, R. (2006). The strong perfect graph theorem. *Ann. Math*, *164*, 51–229.

Chvátal, V. (1985). Star-cutsets and perfect graphs. *J. Comb. Theory, Ser. B*, *39*, 189–199.

Cooper, M. C., de Givry, S., Sanchez, M., Schiex, T., Zytnicki, M., & Werner, T. (2010). Soft arc consistency revisited. *Artif. Intell.*, *174*, 449–478.

Diestel, R. (2010). *Graph theory*. Springer. Fourth edition.

Faenza, Y., Oriolo, G., & Stauffer, G. (2011). An algorithmic decomposition of claw-free graphs leading to an O(n^3)-algorithm for the weighted stable set problem. *SODA* (pp. 630–646).

Foulds, J., Navaroli, N., Smyth, P., & Ihler, A. (2011). Revisiting MAP estimation, message passing, and perfect graphs. *Artificial Intelligence and Statistics.*

Gallai, T. (1962). Graphen mit triangulierbaren ungeraden Vielecken. *Magyar Tud. Akad. Mat. Kutató Int. Közl.*, *7*, 3–36.

Greig, D., Porteous, B., & Seheult, A. (1989). Exact maximum a posteriori estimation for binary images. *J. Royal Statistical Soc., Series B*, *51*, 271–279.

Grötschel, M., Lovász, L., & Schrijver, A. (1984). *Topics on perfect graphs*, chapter Polynomial algorithms for perfect graphs. North-Hollannd, Amsterdam.

Harary, F. (1953). On the notion of balance of a signed graph. *Michigan Mathematical Journal*, *2*, 143–146.

Huang, B., & Jebara, T. (2007). Loopy belief propagation for bipartite maximum weight b-matching. *Artificial Intelligence and Statistics.*

Jebara, T. (2009). MAP estimation, message passing, and perfect graphs. *Uncertainty in Artificial Intelligence.*

Jebara, T. (2012). *Tractability: Practical approaches to hard problems*, chapter Perfect graphs and graphical modeling. Cambridge Press.

Jégou, P. (1993). Decomposition of domains based on the micro-structure of finite constraint-satisfaction problems. *AAAI* (pp. 731–736).

Kolmogorov, V., & Zabih, R. (2004). What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Analysis and Machine Intelligence*, *26*, 147–159.

Lovász, L. (1972). Normal hypergraphs and the perfect graph conjecture. *Discrete Mathematics*, *2*, 253–267.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference.* Morgan Kaufmann.

Pletscher, P., & Kohli, P. (2012). Learning low-order models for enforcing high-order statistics. *AISTATS.*

Ravikumar, P., & Lafferty, J. (2006). Quadratic programming relaxations for metric labeling and Markov random field MAP estimation. *International Conference on Machine Learning.*

Rother, C., Kolmogorov, V., Lempitsky, V. S., & Szummer, M. (2007). Optimizing binary MRFs via extended roof duality. *CVPR.*

Sanghavi, S., Malioutov, D., & Willsky, A. (2008). Linear programming analysis of loopy belief propagation for weighted matching. *Neural Information Processing Systems.*

Sanghavi, S., Shah, D., & Willsky, A. S. (2009). Message passing for maximum weight independent set. *IEEE Transactions on Information Theory*, *55*, 4822–4834.

Shimony, S. (1994). Finding MAPs for belief networks is NP-hard. *Aritifical Intelligence*, *68*, 399–410.

Sontag, D., Meltzer, T., Globerson, A., Jaakkola, T., & Weiss, Y. (2008). Tightening LP relaxations for MAP using message passing. *UAI* (pp. 503–510). AUAI Press.

Wainwright, M., & Jordan, M. (2008). Graphical models, exponential families and variational inference. *Foundations and Trends in Machine Learning*, *1*, 1–305.

Weiss, Y., Yanover, C., & Meltzer, T. (2007). MAP estimation, linear programming and belief propagation with convex free energies. *UAI* (pp. 416–425). AUAI Press.

Yedidia, J., Freeman, W., & Weiss, Y. (2001). Understanding belief propagation and its generalizations. *International Joint Conference on Artificial Intelligence, Distinguished Lecture Track.*

Yildirim, E., & Fan-Orzechowski, X. (2006). On extracting maximum stable sets in perfect graphs using Lovász's theta function. *Computational Optimization and Applications*, *33*, 229–247.

Zivny, S., Cohen, D. A., & Jeavons, P. G. (2009). The expressive power of binary submodular functions. *Discrete Applied Mathematics*, *157*, 3347–3358.

Integrating Document Clustering and Topic Modeling

Pengtao Xie[*]
State Key Laboratory on Intelligent Technology and Systems
Tsinghua National Lab for Information Science and Technology
Department of Computer Science and Technology
Tsinghua University, Beijing 100084, China

Eric P.Xing
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Abstract

Document clustering and topic modeling are two closely related tasks which can mutually benefit each other. Topic modeling can project documents into a topic space which facilitates effective document clustering. Cluster labels discovered by document clustering can be incorporated into topic models to extract local topics specific to each cluster and global topics shared by all clusters. In this paper, we propose a multi-grain clustering topic model (MGCTM) which integrates document clustering and topic modeling into a unified framework and jointly performs the two tasks to achieve the overall best performance. Our model tightly couples two components: a mixture component used for discovering latent groups in document collection and a topic model component used for mining multi-grain topics including local topics specific to each cluster and global topics shared across clusters. We employ variational inference to approximate the posterior of hidden variables and learn model parameters. Experiments on two datasets demonstrate the effectiveness of our model.

1 INTRODUCTION

In the text domain, document clustering (Aggarwal and Zhai, 2012; Cai et al., 2011; Lu et al., 2011; Ng et al., 2002; Xu and Gong, 2004; Xu et al., 2003) and topic modeling (Blei et al., 2003; Hofmann, 2001) are two widely studied problems which have many applications. Document clustering aims to organize similar documents into groups, which is crucial for document organization, browsing, summarization, classification and retrieval. Topic modeling develops probabilistic generative models to discover the latent semantics embedded in document collection and has demonstrated vast success in modeling and analyzing texts.

Document clustering and topic modeling are highly correlated and can mutually benefit each other. On one hand, topic models can discover the latent semantics embedded in document corpus and the semantic information can be much more useful to identify document groups than raw term features. In classic document clustering approaches, documents are usually represented with a bag-of-words (BOW) model which is purely based on raw terms and is insufficient to capture all semantics. Topic models are able to put words with similar semantics into the same group called topic where synonymous words are treated as the same. Under topic models, document corpus is projected into a topic space which reduces the noise of similarity measure and the grouping structure of the corpus can be identified more effectively.

On the other hand, document clustering can facilitate topic modeling. Specifically, document clustering enables us to extract local topics specific to each document cluster and global topics shared across clusters. In a collection, documents usually belong to several groups. For instance, in scientific paper archive such as Google Scholar, papers are from multiple disciplines, such as math, biology, computer science, economics. Each group has its own set of topics. For instance, computer science papers cover topics like operating system, network, machine learning while economics papers contain topics like entrepreneurial economics, financial economics, mathematical economics. Besides group-specific topics, a common set of global topics are shared by all groups. In paper archive, papers from all groups share topics like reviewing related work, reporting experimental results and acknowledging financial supports. Clustering can help us to identify the latent groups in a document collection and subsequently we can identify local topics specific to each group and

[*]The work is done while Pengtao Xie was visiting Carnegie Mellon University.

global topics shared by all groups by exploiting the grouping structure of documents. These fine-grained topics can facilitate a lot of utilities. For instance, we can use the group-specific local topics to summarize and browser a group of documents. Global topics can be used to remove background words and describe the general contents of the whole collection. Standard topic models (Blei et al., 2003; Hofmann, 2001) lack the mechanism to model the grouping behavior among documents, thereby they can only extract a single set of flat topics where local topics and global topics are mixed and can not be distinguished.

Naively, we can perform these two tasks separately. To make topic modeling facilitates clustering, we can first use topic models to project documents into a topic space, then perform clustering algorithms such as K-means in the topic space to obtain clusters. To make clustering promotes topic modeling, we can first obtain clusters using standard clustering algorithms, then build topic models to extract cluster-specific local topics and cluster-independent global topics by incorporating cluster labels into model design. However, this naive strategy ignores the fact that document clustering and topic modeling are highly correlated and follow a chicken-and-egg relationship. Better clustering results produce better topic models and better topic models in turn contribute to better clustering results. Performing them separately fails to make them mutually promote each other to achieve the overall best performance.

In this paper, we propose a generative model which integrates document clustering and topic modeling together. Given a corpus, we assume there exist several latent groups and each document belongs to one latent group. Each group possesses a set of local topics that capture the specific semantics of documents in this group and a Dirichlet prior expressing preferences over local topics. Besides, we assume there exist a set of global topics shared by all groups to capture the common semantics of the whole collection and a common Dirichlet prior governing the sampling of proportion vectors over global topics for all documents. Each document is a mixture of local topics and global topics. Words in a document can be either generated from a global topic or a local topic of the group to which the document belongs. In our model, the latent variables of cluster membership, document-topic distribution and topics are jointly inferred. Clustering and modeling are seamlessly coupled and mutually promoted.

The major contribution of this paper can be summarized as follows

- We propose a unified model to integrate document clustering and topic modeling together.
- We derive variational inference for posterior inference and parameter learning.
- Through experiments on two datasets, we demonstrate the capability of our model in simultaneously clustering document and extracting local and global topics.

The rest of this paper is organized as follows. Section 2 reviews related work. In Section 3, we propose the MGCTM model and present a variational inference method. Section 4 gives experimental results. Section 5 concludes the paper and points out future research directions.

2 RELATED WORK

2.1 DOCUMENT CLUSTERING

Document clustering (Aggarwal and Zhai, 2012; Cai et al., 2011; Lu et al., 2011; Ng et al., 2002; Xu and Gong, 2004; Xu et al., 2003) is a widely studied problem with many applications such as document organization, browsing, summarization, classification. See (Aggarwal and Zhai, 2012) for a broad overview. Popular clustering methods such as K-means and spectral clustering (Ng et al., 2002; Shi and Malik, 2000) in general clustering literature are extensively used for document grouping.

Specific to text domain, one popular paradigm of clustering methods is based on matrix factorization, including Latent Semantic Indexing (LSI) (Deerwester et al., 1990), Non-negative Matrix Factorization (NMF) (Xu et al., 2003) and Concept Factorization (Cai et al., 2011; Xu and Gong, 2004). The basic idea of factorization based methods is to transform documents from the original term space to a latent space. The transformation can reduce data dimensionality, reduce the noise of similarity measure and magnify the semantic effects in the underlying data (Aggarwal and Zhai, 2012), which are beneficial for clustering.

Researchers have applied topic models to cluster documents. (Lu et al., 2011) investigated clustering performance of PLSA and LDA. They use LDA and PLSA to model the corpus and each topic is treated as a cluster. Documents are clustered by examining topic proportion vector $\boldsymbol{\theta}$. A document is assigned to cluster x if $x = \text{argmax}_j \theta_j$.

2.2 TOPIC MODELING

Topic models (Blei et al., 2003; Hofmann, 2001) are probabilistic generative models initially created to model texts and identify latent semantics underlying document collection. Topic models posit document collection exhibits multiple latent semantic topics where each topic is represented as a multinomial distribution over a given vocabulary and each document is a mixture of hidden topics. In the vision domain, topic models (Fei-Fei and Perona, 2005; Zhu et al., 2010) are also widely used for image modeling.

Several models have been devised to jointly model data and their category labels or cluster labels. Fei-Fei (Fei-Fei and Perona, 2005) proposed a Bayesian hierarchical model to jointly model images and their categories. Each category possesses a LDA model with category-specific Dirichlet prior and topics. In their problem, category labels are observed. In this paper, we are interested in unsupervised clustering where cluster label is unknown. Wallach (Wallach, 2008) proposed a cluster based topic model (CTM) which introduces latent variables into LDA to model groups and each group owns a group-specific Dirichlet prior governing the sampling of document-topic distribution. Each document is associated with a group indicator and its topic proportion vector is generated from the Dirichlet prior specific to that group. (Zhu et al., 2010) proposed a similar model used for scene classification in computer vision. They associate each group a logistic-normal prior rather than a Dirichlet prior. However, in the two models, all groups share a single set of topics. They lack the mechanism to identify local topics specific to each cluster and global topics shared by all clusters. Another issue is topics inherently belonging to group A may be used to generate documents in group B, which is problematic. For instance, when modeling scientific papers, it is unreasonable to use a “computer architecture” topic in computer science group to generate an economics paper. Models proposed in (Wallach, 2008; Zhu et al., 2010) can not prohibit this problem since topics are shared across groups. Eventually, the inferred topics will be less coherent and are not discriminative enough to differentiate clusters.

The idea of using fine-grained topics belonging to several sets rather than flat topics from a single set to model documents is exploited in (Ahmed and Xing, 2010; Chemudugunta and Steyvers, 2007; Titov and McDonald, 2008). (Chemudugunta and Steyvers, 2007) represents each document as a combination of a background distribution over common words, a mixture distribution over general topics and a distribution over words that are treated as being specific to that document. (Titov and McDonald, 2008) proposed a multi-grain topic model for online review modeling. They use local topics to capture ratable aspects and utilize global topics to capture properties of reviewed items. (Ahmed and Xing, 2010) proposed a multi-view topic model for ideological perspective analysis. Each ideology has a set of ideology-specific topics and an ideology-specific distribution over words. All documents share a set of ideology-independent topics. In their problem, the ideology label for each document is observed.

3 MULTI-GRAIN CLUSTERING TOPIC MODEL

In this section, we propose the multi-grain clustering topic model (MGCTM) and derive the variational inference method.

3.1 THE MODEL

The MGCTM model is shown in Figure 1. Given a corpus containing N documents $d \in \{1, 2, \cdots, N\}$, we assume these documents inherently belong to J groups $j \in \{1, 2, \cdots, J\}$. Each group j possesses K group-specific local topics $\{\beta_{jk}^{(l)}\}_{k=1}^{K}$. Local topics are used to capture the semantics specific to each group. Besides, each group j has a group-specific local Dirichlet prior $\boldsymbol{\alpha}_j^{(l)}$. Local topic proportion vectors of documents in group j are sampled from $\boldsymbol{\alpha}_j^{(l)}$. Except local topics for each group, we also assume there exist a single set of R global topics $\{\boldsymbol{\beta}_k^{(g)}\}_{k=1}^{R}$ shared by all groups. Global topics are used to model the universal semantics of the whole collection. A global Dirichlet prior $\boldsymbol{\alpha}^{(g)}$ is used to generate proportion vectors over global topics and is shared by all documents. A global multinomial prior $\boldsymbol{\pi}$ is used to choose group membership for a document. π_j denotes the prior probability that a document belongs to group j.

Each document is associated with a group indicator and has a multinomial distribution over local topics and a multinomial distribution over global topics. Words in a document can be either generated from local topics or global topics. We introduce a Bernoulli variable for each word to indicate whether this word is sampled from a global topic or a local topic. The Bernoulli distribution for each document is sampled from a corpus level Beta prior $\boldsymbol{\gamma}$. To generate a document d containing N_d words $\mathbf{w}_d = \{w_i\}_{i=1}^{N_d}$, we first choose a group η_d from the multinomial distribution parametrized by $\boldsymbol{\pi}$. Then from the local Dirichlet prior $\boldsymbol{\alpha}_{\eta_d}^{(l)}$ corresponding to group η_d, we sample a local topic proportion vector $\boldsymbol{\theta}_{\eta_d}^{(l)}$. From the global Dirichlet prior $\boldsymbol{\alpha}^{(g)}$, a multinomial distribution $\boldsymbol{\theta}_d^{(g)}$ over global topics is sampled. From Beta distribution parameterized by $\boldsymbol{\gamma}$, we sample a Bernoulli distribution ω_d from which a binary decision is made at each word position to make choice between local topics and global topics. To gen-

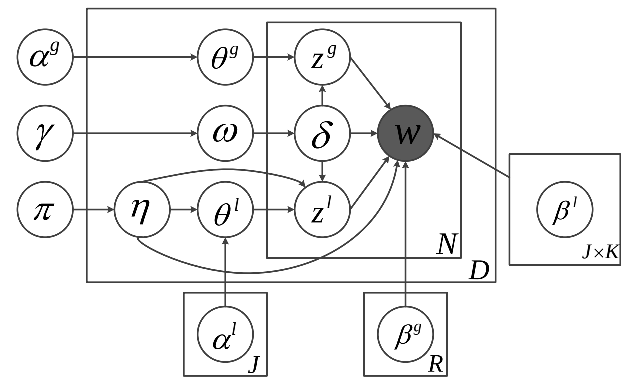


Figure 1: Multi-Grain Clustering Topic Model (MGCTM)

erate a word w_{di}, we first pick a binary variable δ_{di} from the Bernoulli distribution parameterized by ω_d. If $\delta_{di} = 1$, we assume w_{di} is generated from a local topic. A local topic $z^{(l)}_{\eta_d,i}$ is picked up from the local topic proportion vector $\boldsymbol{\theta}^{(l)}_{\eta_d}$ and w_{di} is generated from the topic-word distribution corresponding to local topic $z^{(l)}_{di}$ and group η_d. If $\delta_{di} = 0$, we assume w_{di} is generated from a global topic. In this case, a global topic $z^{(g)}_{di}$ is first picked up from the global topic proportion vector $\boldsymbol{\theta}^{(g)}_{d}$ and w_{di} is generated from the topic-word distribution corresponding to global topic $z^{(g)}_{di}$.

The generative process of a document in MGCTM can be summarized as follows

- Sample a group $\eta \sim Multi(\boldsymbol{\pi})$
- Sample local topic proportion $\boldsymbol{\theta}^{(l)}_{\eta} \sim Dir(\alpha^{(l)}_{\eta})$
- Sample global topic proportion $\boldsymbol{\theta}^{(g)} \sim Dir(\alpha^{(g)})$
- Sample Bernoulli parameter $\omega \sim Beta(\boldsymbol{\gamma})$
- For each word w
  - Sample a binary indicator $\delta \sim Bernoulli(\omega)$
  - If $\delta = 1$
    - sample a local topic $z^{(l)}_{\eta} \sim Multi(\boldsymbol{\theta}^{(l)}_{\eta})$
    - sample $w \sim Multi(\boldsymbol{\beta}_{z^{(l)}_{\eta}})$
  - If $\delta = 0$
    - sample a global topic $z^{(g)} \sim Multi(\boldsymbol{\theta}^{(g)})$
    - sample $w \sim Multi(\boldsymbol{\beta}_{z^{(g)}})$

We claim that performing document clustering and modeling jointly is superior to doing them separately. MGCTM consists of a mixture model component and a topic model component. Document clustering is accomplished by estimating ζ and $\boldsymbol{\pi}$ of the mixture component. Topic modeling involves inferring ω, $\boldsymbol{\Theta}^{(l)}$, $\boldsymbol{\theta}^{(g)}$, $\boldsymbol{\delta}$, $\mathbf{Z}^{(l)}$, $\mathbf{z}^{(g)}$, $\boldsymbol{\gamma}$, $\mathbf{A}^{(l)}$, $\boldsymbol{\alpha}^{(g)}$,$\mathbf{B}^{(l)}$, $\mathbf{B}^{(g)}$ of the topic model component. As described in Section 3.2, latent variables are inferred by maximizing the log likelihood of observed data $\{\mathbf{w}_d\}_{d=1}^{D}$ or its lower bound. Performing clustering and modeling separately is equivalent to inferring latent variables of one component while fixing those of the other component. In the case where we first fit documents using topic model and then perform clustering, we are actually clamping the latent variables of topic model component in MGCTM to some predefined values and then estimating the mixture model component by maximizing the log likelihood (or its lower bound) of observations. In the other case where topic modeling follows clustering, latent variables of mixture model component are predefined and we maximize the log likelihood (or its lower bound) only with respect to those of the topic model component. In contrast, performing the two tasks jointly is equivalent to maximizing the log likelihood (or its lower bound) w.r.t latent variables of two components simultaneously. Suppose we aim to maximize a function $f(\mathbf{x})$ defined over $\mathbf{x}$. $\mathbf{x}$ can be partitioned into two subsets $\mathbf{x}_A$ and $\mathbf{x}_B$. Let $f(\mathbf{x}^*)$ denote the optimal value that can be achieved over $\mathbf{x}$. Let $f(\mathbf{x}^*_A, \mathbf{x}_B = \mathbf{c})$ denote the optimal value obtained by optimizing $\mathbf{x}_A$ while fixing $\mathbf{x}_B$ to some preset value $\mathbf{c}$. Let $f(\mathbf{x}^*_B, \mathbf{x}_A = \mathbf{d})$ denote the optimal value obtained by optimizing $\mathbf{x}_B$ while fixing $\mathbf{x}_A$ to some preset value $\mathbf{d}$. Clearly, the following inequalities hold: $f(\mathbf{x}^*) \geq f(\mathbf{x}^*_A, \mathbf{x}_B = \mathbf{c})$, $f(\mathbf{x}^*) \geq f(\mathbf{x}^*_B, \mathbf{x}_A = \mathbf{d})$. From this property, we can conclude that jointly performing clustering and modeling grants us better results than doing them separately.

It would be interesting to make a comparison of our model with Gaussian mixture model (GMM) and cluster based topic models (CTM) (Wallach, 2008; Zhu et al., 2010) in the context of document clustering and modeling. In GMM, each document is converted into a term vector. GMM associates each cluster a multivariate Gaussian distribution. To generate a document, GMM first samples a cluster, then generate the document from the Gaussian distribution corresponding to this cluster. In contrast, our model is a mixture of LDAs. Each cluster is characterized by a LDA model with a set of topics specific to this cluster and a unique Dirichlet prior from which document-topic distributions are sampled. To generate a document, our model first samples a cluster, then use the corresponding LDA to generate the document. In GMM, documents are represented with raw terms, which are insufficient to capture underlying semantics. In our model, documents are modeled using LDA, which is well-known for its capability to discover latent semantics. Different from CTM (Wallach, 2008; Zhu et al., 2010) where all LDAs share a common set of topics, we allocate each LDA a set of topics in our model. This specific design owns two advantages. First, it can explicitly infer group-specific topics for each cluster. Second, it can avoid the problem of using topics of one group to generate documents in another group.

3.2 VARIATIONAL INFERENCE AND PARAMETER LEARNING

The key inference problem involved in our model is to estimate the posterior distribution $p(\eta, \omega, \mathbf{\Theta}^{(l)}, \boldsymbol{\theta}^{(g)}, \boldsymbol{\delta}, \mathbf{Z}^{(l)}, \mathbf{z}^{(g)} | \mathbf{w}, \mathbf{\Theta})$ of latent variables $\mathbf{H} = \{\eta, \omega, \mathbf{\Theta}^{(l)}, \boldsymbol{\theta}^{(g)}, \boldsymbol{\delta}, \mathbf{Z}^{(l)}, \mathbf{z}^{(g)}\}$ given observed variables $\mathbf{w}$ and model parameters $\mathbf{\Pi} = \{\boldsymbol{\pi}, \boldsymbol{\gamma}, \mathbf{A}^{(l)}, \boldsymbol{\alpha}^{(g)}, \mathbf{B}^{(l)}, \mathbf{B}^{(g)}\}$. Since extract inference is intractable, we use variational inference (Wainwright and Jordan, 2008) to approximate the posterior. The basic idea is to employ another distribution $q(\mathbf{H}|\mathbf{\Omega})$ which is parametrized by $\mathbf{\Omega}$ and approximate the true posterior by minimizing the Kullback-Leibler (KL) divergence between $p(\mathbf{H}|\mathbf{w}, \mathbf{\Pi})$ and $q(\mathbf{H}|\mathbf{\Omega})$, which is equivalent to maximizing a lower bound $\mathbb{E}_q[\log p(\mathbf{H}, \mathbf{w}|\mathbf{\Pi})] - \mathbb{E}_q[\log q(\mathbf{H}|\mathbf{\Omega})]$ of data likelihood. The maximization is achieved via an iterative fixed-point method. In E-step, the model parameters $\mathbf{\Pi}$ is fixed and we update the variational parameters $\mathbf{\Omega}$ by maximizing the lower bound. In M-step, we fix the variational parameters and update the model parameters. This process continues until convergence.

The variational distribution q is defined as follows

$$\begin{aligned} & q(\eta, \omega, \mathbf{\Theta}^{(l)}, \boldsymbol{\theta}^{(g)}, \boldsymbol{\delta}, \mathbf{Z}^{(l)}, \mathbf{z}^{(g)}) \\ & = q(\eta|\boldsymbol{\zeta}) q(\omega|\boldsymbol{\lambda}) \prod_{j=1}^{J} q(\boldsymbol{\theta}_j^{(l)}|\boldsymbol{\mu}_j^{(l)}) q(\boldsymbol{\theta}^{(g)}|\boldsymbol{\mu}^{(g)}) \\ & \prod_{i=1}^{N} q(\delta_i|\tau_i) \prod_{j=1}^{J} q(z_{i,j}^{(l)}|\boldsymbol{\phi}_{i,j}^{(l)}) q(z_i^{(g)}|\boldsymbol{\phi}_i^{(g)}) \end{aligned} \tag{1}$$

where $\boldsymbol{\zeta}$, $\{\boldsymbol{\phi}_{i,j}^{(l)}\}_{i=1,j=1}^{i=N,j=J}$ and $\{\boldsymbol{\phi}_i^{(g)}\}_{i=1}^{N}$ are multinomial parameters, $\boldsymbol{\lambda}$ is Beta parameter, $\boldsymbol{\mu}^{(l)}$ and $\boldsymbol{\mu}^{(g)}$ are Dirichlet parameters, $\{\tau_i\}_{i=1}^{N}$ are Bernoulli parameters.

In E-step, we compute the variational parameters while keeping model parameters fixed

$$\begin{aligned} & \zeta_j \propto \pi_j \exp\{\log \Gamma(\sum_{i=1}^{K} \alpha_{ji}^{(l)}) - \sum_{i=1}^{K} \log \Gamma(\alpha_{ji}^{(l)}) \\ & + \sum_{k=1}^{K} (\alpha_{jk}^{(l)} - 1)(\Psi(\mu_{jk}^{(l)}) - \Psi(\sum_{i=1}^{K} \mu_{ji}^{(l)})) \\ & + \sum_{i=1}^{N} \tau_i \{ \sum_{k=1}^{K} \phi_{i,j,k}^{(l)} (\Psi(\mu_{jk}^{(l)}) - \Psi(\sum_{n=1}^{K} \mu_{j,n}^{(l)})) \\ & + \sum_{k=1}^{K} \sum_{v=1}^{V} \phi_{i,j,k}^{(l)} w_{iv} \log \beta_{j,k,v}^{(l)} \} \} \end{aligned} \tag{2}$$

$$\lambda_1 = \gamma_1 + \sum_{i=1}^{N} \tau_i, \lambda_2 = \gamma_2 + \sum_{i=1}^{N} (1 - \tau_i) \tag{3}$$

$$\mu_{jk}^{(l)} = \zeta_j \alpha_{jk}^{(l)} + \sum_{i=1}^{N} \tau_i \zeta_j \phi_{i,j,k}^{(l)} + 1 - \zeta_j \tag{4}$$

$$\mu_k^{(g)} = \alpha_k^{(g)} + \sum_{i=1}^{N} (1 - \tau_i) \phi_{ik}^{(g)} \tag{5}$$

$$\begin{aligned} & \tau_i = \{1 + \exp\{-\Psi(\gamma_1) + \Psi(\gamma_2) \\ & - \sum_{j=1}^{J} \sum_{k=1}^{K} \zeta_j \phi_{i,j,k}^{(l)} (\Psi(\mu_{jk}^{(l)}) - \Psi(\sum_{n=1}^{K} \mu_{jn}^{(l)})) \\ & - \sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{v=1}^{V} \zeta_j \phi_{i,j,k}^{(l)} w_{iv} \log \beta_{j,k,v}^{(l)} \\ & + \sum_{k=1}^{K} \phi_{ik}^{(g)} (\Psi(\mu_k^{(g)}) - \Psi(\sum_{j=1}^{K} \mu_j^{(g)})) \\ & + \sum_{k=1}^{K} \sum_{v=1}^{V} \phi_{ik}^{(g)} w_{iv} \log \beta_{k,v}^{(g)} \} \}^{-1} \end{aligned} \tag{6}$$

$$\begin{aligned} & \phi_{i,j,k}^{(l)} \propto \exp\{\tau_i \zeta_j (\Psi(\mu_{jk}^{(l)}) - \Psi(\sum_{n=1}^{K} \mu_{j,n}^{(l)}) \\ & + \sum_{v=1}^{V} w_{iv} \log \beta_{j,k,v}^{(l)})\} \end{aligned} \tag{7}$$

$$\begin{aligned} & \phi_{ik}^{(g)} \propto \exp\{(1 - \tau_i)(\Psi(\mu_k^{(g)}) - \Psi(\sum_{j=1}^{K} \mu_j^{(g)}) \\ & + \sum_{v=1}^{V} w_{iv} \log \beta_{k,v}^{(g)})\} \end{aligned} \tag{8}$$

In M-step, we optimize the model parameters by maximizing the lower bound

$$\pi_j = \frac{\sum_{d=1}^{D} \zeta_{dj}}{D} \tag{9}$$

$$\beta_{j,k,v}^{(l)} \sim \sum_{d=1}^{D} \sum_{i=1}^{N_d} \zeta_j \tau_{di} \phi_{d,i,j,k}^{(l)} w_{d,i,v} \tag{10}$$

$$\beta_{k,v}^{(g)} \sim \sum_{d=1}^{D} \sum_{i=1}^{N_d} (1 - \tau_{di}) \phi_{d,i,k}^{(g)} w_{d,i,v} \tag{11}$$

We optimize Dirichlet priors $\mathbf{A}^{(l)}$, $\boldsymbol{\alpha}^{(g)}$ and Beta priors $\boldsymbol{\gamma}$ using the Newton-Raphson method described in (Blei et al., 2003).

4 EXPERIMENTS

We evaluate the document clustering performance of our model and corroborate its ability to mine group-specific local topics and group-independent global topics on two datasets.

4.1 DOCUMENT CLUSTERING

We evaluate the document clustering performance of our method in this section.

4.1.1 Datasets

The experiments are conducted on Reuters-21578 and 20-Newsgroups datasets. These two datasets are the most widely used benchmark in document clustering. For Reuters-21578, we only retain the largest 10 categories and discard documents with more than one labels, which left us with 7,285 documents. 20-Newsgroups dataset contains 18,370 documents from 20 groups. In all corpus, the stop words are removed and each document is represented as a tf-idf vector.

4.1.2 Experimental Settings

Following (Cai et al., 2011), we use two metrics to measure the clustering performance: accuracy (AC) and normalized mutual information (NMI). Please refer to (Cai et al., 2011) for definitions of these two metrics.

We compare our method with the following baseline methods: K-means (KM) and Normalized Cut (NC) which are probably the most widely used clustering algorithms; Non-negative Matrix Factorization (NMF), Latent Semantic Indexing (LSI), Locally Consistent Concept Factorization (LCCF) which are factorization based approaches showing great effectiveness for clustering documents. To study how topic modeling can affects document clustering, we compare with three topic model based methods. The first one is a naive approach which first uses LDA to learn a topic proportion vector for each document, then performs K-means on topic proportion vectors to obtain clusters. We use LDA+Kmeans to denote this approach. The second one is proposed in (Lu et al., 2011), which treats each topic as a cluster. Document-topic distribution $\boldsymbol{\theta}$ can be deemed as a mixture proportion vector over clusters and can be utilized for clustering. A document is assigned to cluster x if $x = \text{argmax}_j \theta_j$. Note that this approach is a naive solution for integrating document clustering and modeling together. We use LDA+Naive to denote this approach. The third one is cluster based topic model (CTM) (Wallach, 2008) which integrates document clustering and modeling as a whole.

In our experiments, the input cluster number required by clustering algorithms is set to the ground truth number of categories in corpus. Hyperparameters are tuned to achieve the best clustering performance. In NC, we use Gaussian kernel as similarity measure between documents. The bandwidth parameter is set to 10. In LSI, we retain top 300 eigenvectors to form the new subspace. The parameters of LCCF are set as those suggested in (Cai et al., 2011). In LDA+Kmeans and LDA+Naive, we use symmetric Dirichlet prior α and β to draw document-topic distribution and topic-word distribution. α and β are set to 0.1 and 0.01 respectively. In LDA+Kmeans, the number of topics is set to 60. In CTM, we set the number of topics to 60 for Reuters-21578 and 120 for 20-Newsgroups. For MGCTM, we set 5 local topics for each cluster and 10 global topics in Reuters-21578 dataset and 10 local topics for each cluster and 20 global topics for 20-Newsgroups dataset. In MGCTM, we initialize ζ with clustering results obtained from LDA+Naive. The other parameters are initialized randomly.

Table 1: Clustering Accuracy (%)

	Reuters-21578	20-Newsgroups
KM	35.02	33.65
NC	26.22	22.03
NMF	49.58	31.85
LSI	42.00	32.33
LCCF	33.07	11.71
LDA+Kmeans	29.73	37.19
LDA+Naive	54.88	55.38
CTM	**56.58**	45.63
MGCTM	56.01	**58.69**

Table 2: Normalized Mutual Information (%)

	Reuters-21578	20-Newsgroups
KM	35.76	31.54
NC	27.40	20.31
NMF	35.89	27.82
LSI	37.14	29.78
LCCF	30.45	11.40
LDA+Kmeans	36.00	38.15
LDA+Naive	48.00	57.21
CTM	46.52	51.63
MGCTM	**50.10**	**61.59**

4.1.3 Results

Table 1 and Table 2 summarize the accuracy and normalized mutual information of different clustering methods, respectively. It can be seen that topic modeling based clustering methods including LDA+Kmeans, LDA+Naive, CTM and MGCTM are generally better than K-means, normalized cut and factorization based methods. This corroborates our assumption that topic modeling can promote document clustering. The semantics discovered by topic models can effectively facilitate accurate similarity measure, which is helpful to obtain coherent clusters.

Compared with LDA+Kmeans which performing clustering and modeling separately, three methods including LDA+Naive, CTM and MGCTM which jointly performing two tasks achieve much better results. This corroborates our assumption that clustering and modeling can mutually promote each other and couple them into a unified framework produces superior performance than separating them into two procedures.

Among LDA+Naive, CTM and MGCTM which unify clustering and modeling, our approach is generally better than or comparable with the other two. This is because MGCTM possesses more sophistication in terms of model design, which in turn contributes to better clustering results. LDA+Naive assigns each cluster only one topic, which may not be sufficient to capture the diverse semantics within each cluster. CTM fails to differentiate cluster-specific topics and cluster-independent topics, thereby, the learned topics are not discriminative in distinguishing clusters. Since topics are shared by all clusters, CTM may try to use a topic inherently belonging to cluster A to model a document in cluster B, which is unreasonable and can cause semantic confusion. Our model assigns each cluster a set of topics and can avoid to use topics from one cluster to model documents in another cluster, which is more suitable to produce coherent clusters.

4.2 TOPIC MODELING

In this section, we study the topic modeling capability of our model. We compare with two methods. The first one is a naive approach which first uses K-means to obtain document clusters, then clamps the values of document membership variables ζ in MGCTM to the obtained clusters labels and learns the latent variables corresponding to topic model component. We use Kmeans+MGCTM to denote this approach. Again, the purpose of comparing with this naive approach is to investigate whether integrating clustering and modeling together is superior to doing them separately. The other approach is CTM (Wallach, 2008). We use three models to fit the 20-Newsgroups dataset. The reason to choose 20-Newsgroups rather than Reuters-21578 for topic modeling evaluation is that the categories in 20-Newsgroups are more semantically clear than those in Reuters-21578. In CTM, we set the topic number to 120. In MGCTM and Kmeans+MGCTM, we set 5 local topics for each of the 20 groups and set 20 global topics. We evaluate the inferred topics both qualitatively and quantitatively. Specifically, we are interested in two things. First, how coherent a topic (either local topic or global topic) is. Second, how is a local topic related to a cluster.

4.2.1 Qualitative Evaluation

Table 3 shows three global topics inferred from 20-Newsgroups by MGCTM. Each topic is represented by the ten most probable words for that topic. It can be seen that these global topics capture the common semantics in the whole corpus and is not specifically associated with a certain news group. Global topic 9 is about news archive organization. Topic 10 is about time. Topic 19 is about article writing. These topics can be used to generate documents in all groups.

Table 3: Three Global Topics Inferred from 20-Newsgroups by MGCTM

Topic 9	Topic 10	Topic 19
section	time	introduction
set	year	information
situations	period	archive
volume	full	address
sets	local	articles
field	future	press
situation	note	time
select	meet	body
hand	case	text
designed	setting	list

Table 4 shows local topics for 4 obtained clusters[1]. As can be seen, local topics effectively capture the specific semantics of each cluster. For instance, in Cluster 1, all the four local topics are highly related with computer, including server, program, Windows, display. In Cluster 2, all topics are about middle east politics, including race, war, religion, diplomacy. In Cluster 3, all topics are about space technology, including space, planets, spacecraft, NASA. In Cluster 4, all topics are closely related with health, including disease, patients, doctors, food. These local topics enable us to understand each cluster easily and clearly, without the burden of browsing a number of documents in a cluster. In our model, documents in a cluster can only be generated from local topics of that cluster and we prohibit to use local topics of cluster A to generate documents in cluster B. Thereby, each local topic is highly related with its own cluster and has almost no correlation with other clusters. In other words, the leaned local topics are very discriminative to differentiate clusters. On the contrary, topics in CTM are shared by all groups. Consequently, the semantic meaning of a topic is very ambiguous and the topic can be related with multiple clusters simultaneously. These topics are suboptimal to summarize clusters because of their vagueness. In Kmeans+MGCTM, the clusters are predefined using K-means, whose clustering performance is much worse than MGCTM as reported in Section 4.1.3. As a result, the quality of learned topics by Kmeans+MGCTM is also worse than MGCTM. Their

[1]Due to space limit, we only show four local topics for each cluster.

Table 4: Lobal Topics of 4 Clusters Inferred from 20-Newsgroups by MGCTM

Cluster 1				Cluster 2			
Topic 1	Topic 2	Topic 3	Topic 4	Topic 1	Topic 2	Topic 3	Topic 4
sun	window	server	motif	muslims	armenian	turkish	armenian
file	manager	lib	file	serbs	azerbaijan	turkey	armenians
openwindows	display	file	version	bosnian	people	university	turkish
xview	event	xfree	mit	bosnia	armenia	history	armenia
echo	motif	xterm	color	henrik	armenians	kuwait	people
usr	application	running	font	war	turkish	jews	genocide
xterm	program	mit	server	armenians	azeri	people	turks
display	widget	usr	sun	muslim	soviet	professor	soviet
ftp	win	window	fonts	turkey	dead	government	war
run	screen	clients	tar	world	russian	turks	russian
Cluster 3				**Cluster 4**			
Topic 1	Topic 2	Topic 3	Topic 4	Topic 1	Topic 2	Topic 3	Topic 4
space	space	nasa	space	candida	people	vitamin	msg
nasa	launch	gov	nasa	yeast	pitt	cancer	food
hst	cost	space	apr	weight	chronic	medical	people
larson	shuttle	energy	alaska	patients	evidence	information	time
mission	dc	apr	earth	doctor	body	disease	foods
orbit	station	earth	satellite	lyme	time	treatment	chinese
theory	nuclear	ca	gov	disease	disease	patients	eat
universe	power	jpl	people	kidney	medicine	retinol	good
light	program	higgins	high	good	years	good	pain
mass	system	gary	shuttle	people	water	pms	effects

quantitative comparison is reported in Section 4.2.2.

4.2.2 Quantitative Evaluation

How to quantitatively evaluate topic models is a open problem (Boyd-Graber et al., 2009). Some researchers resort to perplexity or held-out likelihood. Such measures are useful for evaluating the predictive model (Boyd-Graber et al., 2009). However, they are not capable to evaluate how coherent and meaningful the inferred topics are. Through large-scale user studies, (Boyd-Graber et al., 2009) shows that topic models which perform better on held-out likelihood may infer less semantically meaningful topics. Thereby, we do not use perplexity or held-out likelihood as evaluation metric.

To evaluate how coherent a topic is, we pick up top 20 candidate words for each topic and ask 5 student volunteers to label them. First, the volunteers need to judge whether a topic is interpretable or not. If not, the 20 candidate words in this topic are automatically labeled as "irrelevant". Otherwise, volunteers are asked to identify words that are relevant to this topic. Coherence measure (CM) is defined as the ratio between the number of relevant words and total number of candidate words.

Table 5 summarizes the coherence measure collected from 5 students. As can be seen, the average coherence of topics inferred by our model surpasses those learned from Kmeans+MGCTM and CTM. In our model, background words in the corpus are organized into global topics and words specific to clusters are mapped into local topics. Kmeans+MGCTM learns local topics based on the cluster labels obtained by K-means. Due to the suboptimal clustering performance of K-means, some documents similar in semantics are put into different clusters while some dissimilar documents are put into the same cluster. Consequently, the learned local topics are less reasonable since they are resulted from poor cluster labels. CTM lack the mechanism to differentiate corpus-level background words and cluster-specific words and these two types of words are mixed in many topics, making topics hard to interpret and less coherent.

Table 5: Coherence Measure (CM) (%) of Learned Topics

	Kmeans+MGCTM	CTM	MGCTM
annotator 1	30.17	28.88	**36.08**
annotator 2	36.50	43.54	**45.79**
annotator 3	29.38	**35.42**	30.83
annotator 4	18.33	25.96	**29.46**
annotator 5	24.75	24.54	**25.17**
average	27.83	31.60	**33.47**

To measure the relevance between local topics and clusters in our method, from the 5 learned local topics for each cluster, we ask the 5 students to pick up the relevant ones. The relevance measure (RM) is defined as the ratio between number of relevant topics

Table 6: Relevance Measure (RM) (%) between Topics and Clusters

	Kmeans+MGCTM	CTM	MGCTM
annotator 1	64	66	**72**
annotator 2	47	**61**	57
annotator 3	51	54	**63**
annotator 4	76	74	**81**
annotator 5	45	51	**58**
average	56.6	61.2	**66.2**

and total number of topics to be labeled. In CTM, we choose 5 most related topics for each cluster using the method described in (Wallach, 2008) and ask annotators to label.

Table 6 presents the relevance measure between local topics and clusters. The relevance measure in our method is significantly better than Kmeans+MGCTM and CTM. The suboptimal performance of Kmeans+MGCTM still results from the poor clustering performance of K-means. The comparison of Kmeans+MGCTM and MGCTM in Table 5 and Table 6 demonstrates that jointly performing clustering and modeling can produce better local and global topics than performing them separately. In CTM, topics are shared across groups. A certain topic T can be used to model documents belonging to several groups. Consequently T will be a composition of words from multiple groups, making it hard to associate T to a certain group clearly. On the contrary, our model allocates each cluster a set of cluster-specific topics and prohibit to use local topics from one cluster to model documents in another cluster. Thereby the relevance between learned local topics and their clusters can be improved greatly.

5 CONCLUSIONS AND FUTURE WORK

We propose a multi-grain clustering topic model to simultaneously perform document clustering and modeling. Experiments on two datasets demonstrate the fact that these two tasks are closely related and can mutually promote each other. In experiments on document clustering, we show that through topic modeling, clustering performance can be improved. In experiments on topic modeling, we demonstrate that clustering can help infer more coherent topics and can differentiate topics into group-specific ones and group-independent ones.

In future, we will extend our model to semi-supervised clustering settings. In reality, we may have incomplete external knowledge which reveals that some document pairs are likely to be put into the same cluster. How to incorporate these semi-supervised information into our model would be an interesting question.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments and thank Chong Wang, Bin Zhao, Gunhee Kim for their helpful suggestions.

References

Charu C Aggarwal and ChengXiang Zhai. A survey of text clustering algorithms. *Mining Text Data*, pages 77–128, 2012.

Amr Ahmed and Eric P Xing. Staying informed: supervised and semi-supervised multi-view topical analysis of ideological perspective. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1140–1150. Association for Computational Linguistics, 2010.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of Machine Learning Research*, 3:993–1022, 2003.

Jonathan Boyd-Graber, Jordan Chang, Sean Gerrish, Chong Wang, and David Blei. Reading tea leaves: how humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, 2009.

Deng Cai, Xiaofei He, and Jiawei Han. Locally consistent concept factorization for document clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 23(6):902–913, 2011.

Chaitanya Chemudugunta and Padhraic Smyth Mark Steyvers. Modeling general and specific aspects of documents with a probabilistic topic model. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, volume 19, page 241. MIT Press, 2007.

Scott Deerwester, Susan T. Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for Information Science*, 41(6): 391–407, 1990.

Li Fei-Fei and Pietro Perona. A bayesian hierarchical model for learning natural scene categories. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 524–531. IEEE, 2005.

Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1):177–196, 2001.

Yue Lu, Qiaozhu Mei, and ChengXiang Zhai. Investigating task performance of probabilistic topic models: an empirical study of plsa and lda. *Information Retrieval*, 14(2):178–203, 2011.

Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: analysis and an algorithm. *Advances in Neural Information Processing Systems*, 2: 849–856, 2002.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

Ivan Titov and Ryan McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008.

Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.

Hanna M Wallach. Structured topic models for language. *Unpublished doctoral dissertation, Univ. of Cambridge*, 2008.

Wei Xu and Yihong Gong. Document clustering by concept factorization. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 202–209. ACM, 2004.

Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and Development in Informaion Retrieval*, pages 267–273. ACM, 2003.

Jun Zhu, Li-Jia Li, Li Fei-Fei, and Eric P Xing. Large margin learning of upstream scene understanding models. *Advances in Neural Information Processing Systems*, 24, 2010.

Active Learning with Expert Advice

Peilin Zhao
Nanyang Technological University
Singapore 639798
peilinzhao@ntu.edu.sg

Steven C.H. Hoi
Nanyang Technological University
Singapore 639798
chhoi@ntu.edu.sg

Jinfeng Zhuang
Microsoft Corporation
USA 98006
jeffzh@microsoft.com

Abstract

Conventional learning with expert advice methods assume a learner is always receiving the outcome (e.g., class labels) of every incoming training instance at the end of each trial. In real applications, acquiring the outcome from oracle can be costly or time consuming. In this paper, we address a new problem of active learning with expert advice, where the outcome of an instance is disclosed only when it is requested by the online learner. Our goal is to learn an accurate prediction model by asking the oracle the number of questions as small as possible. To address this challenge, we propose a framework of active forecasters for online active learning with expert advice, which attempts to extend two regular forecasters, i.e., Exponentially Weighted Average Forecaster and Greedy Forecaster, to tackle the task of active learning with expert advice. We prove that the proposed algorithms satisfy the Hannan consistency under some proper assumptions, and validate the efficacy of our technique by an extensive set of experiments.

1 Introduction

Learning with expert advice has been extensively studied for years in literature [19, 3, 13, 1]. Typically, a conventional learning with expert advice task assumes an online learner acts in an environment with a pool of experts. At each trial, the learner receives an incoming training instance, and must make a prediction on this instance based on the predictions made by the pool of experts. The outcome of the incoming instance will be disclosed by acquiring the feedback from an oracle after the learner has made the prediction, which in turn determines the incurred losses of the learner and the experts as well. The goal of this problem is to enable the online learner be able to make a prediction as accurate as possible. This framework was first introduced by Littlestone and Warmuth [19], who proposed the well-known weighted majority voting algorithms. Over the past decades, the similar problem has been extensively explored by other studies in literature, including Cesa-Bianchi et al. [3, 2], Freund and Schapire [9], Foster and Vohra [8], Haussler et al. [12], Vovk [22] etc.

The existing learning with expert advice methods assume the outcome (e.g., the true class label) of every incoming training instance will be *always* disclosed from an oracle at each trial. However, requesting the outcome of an instance from the oracle is often expensive or time consuming in many real-world applications. Unlike the conventional approaches, this paper investigates a new framework of active learning with expert advice, in which the outcome of an incoming instance may or may not be disclosed at each trial, depending on if the learner decides to make a request to the oracle. The goal of active learning with expert advice is to train the online learner to make an accurate prediction by making the number of requests to the oracle as small as possible, which is potentially applied for improving online classification with multiple kernel learning [14]. This problem is very challenging because on one hand we need to design an effective strategy to build the online learner for the training instance whenever its outcome is disclosed, and on the other hand, we must decide when the online learner should make a request for an incoming instance.

To overcome the challenge of active learning with expert advice, we present a framework of Active Forecaster algorithms and proposed two specific algorithms: (i) active weighted average forecaster and (ii) active greedy forecaster. We analyze the theoretical regret bound of the proposed algorithms, and validate their empirical efficacy via an extensive set of experiments.

The rest of this paper is organized as follows. Section 2 introduces the problem setting of learning with expert advice and the greedy forecaster algorithm. Section 3 presents the active greedy forecaster algorithm for active learning with expert advice and analyzes its theoretical performance. Section 4 shows our experimental results, and Section 5 concludes this work.

2 Problem Setting and Background

Specifically, we considered solving online classification problem through learning with expert advice. Online classification has been extensively studied in machine learning in the past few years [21, 4, 26, 24] for differen problems, like sentiment detection [17], cost-sensitive classification [23], feature selection [25] and etc. To solve online classification problem, the problem setting of a typical learning with expert advice task is as follows. Consider an unknown sequence of instances $\mathbf{x}_1, \dots, \mathbf{x}_T \in \mathbb{R}^d$, a decision maker termed as "forecaster" aims to predict the outcomes (e.g., class labels) of every incoming instance $\mathbf{x}_t$. The forecaster sequentially computes its predictions based on the predictions from a set of N reference forecasters called as "experts". Specifically, at the t-th round, after receiving an instance $\mathbf{x}_t$, the forecaster first accesses to the predictions made by the set of experts $\{f_{i,t} : \mathbb{R}^d \to [0,1] | i = 1, \dots, N\}$, and then computes its own prediction $p_t \in [0,1]$ based on the predictions of the N experts. After p_t is computed, the true outcome $y_t \in \{0,1\}$ is disclosed.

With the true outcomes revealed from the environment/oracle, the prediction performance of the forecaster and experts can be scored by some nonnegative loss function between p_t and y_t, e.g., the absolute loss that is defined as $\ell(p_t, y_t) = |p_t - y_t|$. We can further calculate the cumulative loss experienced by each expert and the forecaster respectively as follows:

$$L_{i,t} = \sum_{j=1}^{t} \ell(f_i(\mathbf{x}_j), y_j), \quad L_t = \sum_{j=1}^{t} \ell(p_j, y_j)$$

The loss difference between the forecaster and the expert is known as the "regret", i.e.,

$$R_{i,t} = L_t - L_{i,t}, i = 1, \dots, N \tag{1}$$

The goal of learning the forecaster is to make the regret with respect to each expert as small as possible, which is equivalent to minimizing the overall regret R_T, i.e.,

$$R_T = \max_{1 \le i \le N} R_{i,T} = L_T - \min_{1 \le i \le N} L_{i,T} \tag{2}$$

In general, we wish to design an ideal forecaster that can achieve a vanishing per-round regret, i.e.,

$$R_T = o(T) \Longleftrightarrow \lim_{T \to \infty} \frac{1}{T} \left(L_T - \min_{1 \le i \le N} L_{i,T} \right) = 0$$

The above property is known as the Hannan consistency [11]. A forecaster satisfying this property is called a Hannan-consistent forecaster [11, 2].

To solve the above task of learning with expert advice, a natural framework is based on the weighted average prediction strategy. Specifically, at time t, the forecaster makes its own prediction as:

$$p_t = \frac{\sum_{i=1}^{N} w_{i,t-1} f_i(\mathbf{x}_t)}{\sum_{i=1}^{N} w_{i,t-1}} \tag{3}$$

where $w_{i,t-1}$ are the combination weights assigned to the experts at time t-1. The intuitive idea of learning the combination weights is to assign large weights for those experts of low regrets/loss and small weights for those of high regrets/loss.

Next we introduce a special case that leads to the well-known forecaster, called "Exponentially Weighted Average Forecaster" (EWAF). In particular, by choosing $w_{i,t-1} = \exp(\eta L_{i,t-1}) / \sum_{j=1}^{N} \exp(\eta L_{j,t-1})$, where η is a positive parameter, the EWAF strategy makes the following prediction:

$$p_t = \frac{\sum_{i=1}^{N} \exp(-\eta L_{i,t-1}) f_i(\mathbf{x}_t)}{\sum_{i=1}^{N} \exp(-\eta L_{i,t-1})}, \tag{4}$$

In addition to the weighted average forecaster, we also consider another kind of forecaster, known as the "Greedy Forecaster" (GF), which makes the following prediction:

$$p_t = \pi_{[0,1]} \left(\frac{1}{2} + \frac{1}{2\eta} \ln \frac{\sum_{i=1}^{N} \exp(-\eta L_{i,t-1} - \eta \ell(f_i(\mathbf{x}_t), 1))}{\sum_{i=1}^{N} \exp(-\eta L_{i,t-1} - \eta \ell(f_i(\mathbf{x}_t), 0))} \right),$$

where $\pi_{[0,1]}(\cdot) = \max(0, \min(1, \cdot))$. According to the existing studies [2], we have the following theorem of regret bounds of the above EWAF and GF algorithms:

Theorem 1. *Let the loss function $\ell(p, y) = |p - y|$. Then, for any T and $\eta > 0$, and for all $y_1, \dots, y_T \in \{0,1\}$, the regrets of both the EWAF and GF algorithms satisfy*

$$R_T = L_T - \min_{1 \le i \le N} L_{i,T} \le \frac{\ln N}{\eta} + \frac{\eta T}{8}$$

In particular, by choosing $\eta = \sqrt{8 \ln N / T}$, the upper bound of the regret becomes $\sqrt{(T/2) \ln N}$.

The above theorem shows both the EWAF and GF algorithms satisfy the Hannan consistency, i.e., $R_T \le o(T)$, which guarantees that the actual per-round regret R_T / T becomes negligible as T grows.

3 Active Learning with Expert Advice

In this section, we address a new problem of active learning with expert advice. Unlike the above regular learning with expert advice task where the outcome of every incoming instance is *always* revealed to the online learner, in an active learning with expert advice task, the outcome of an incoming instance is *only* revealed whenever the learner has made a request for acquiring the label from the environment/oracle. In this section, we aim to develop a framework of *active forecasters* to tackle the challenging task of active learning with expert advice.

We first introduce binary variables $z_s \in \{0,1\}, s = 1,\ldots,t$ to indicate if an active forecaster has decided to request the class label of an incoming instance received at s-th trial. We denoted by $\widehat{L}_{i,t}$ the loss function experienced by the active learner w.r.t. the ith expert, i.e., $\widehat{L}_{i,t} = \sum_{s=1}^{t} \ell(f_i(\mathbf{x}_s), y_s)z_s$.

Hence, the class label for the t-th example predicted by the active forecaster, denoted by $\widehat{p}_t$, is computed as $\widehat{p}_t = \pi_{[0,1]}(\bar{p}_t)$, where $\bar{p}_t$ is computed by different approaches for different forecasters:

$$\bar{p}_t = \frac{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1}) f_i(\mathbf{x}_t))}{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1})} \text{ (EWAF)};$$

$$\bar{p}_t = \frac{1}{2} + \frac{1}{2\eta} \ln \frac{\sum_{i=1}^{N} \exp[\eta(-\widehat{L}_{i,t-1} - \ell(f_i(\mathbf{x}_t), 1))]}{\sum_{i=1}^{N} \exp[\eta(-\widehat{L}_{i,t-1} - \ell(f_i(\mathbf{x}_t), 0))]} \text{ (GF)}.$$

In the above formula of EWAF, since $\bar{p}_t \in [0,1]$, we always have $\widehat{p}_t = \pi_{[0,1]}(\bar{p}_t) = \bar{p}_t$.

The key challenge for active learning with expert advice is to decide when the active forecaster should or should not make a request to acquire the class label w.r.t. an incoming instance from the environment/oracle. A naive solution is to consider a random sampling approach, which however may not be effective enough (this will be considered as a baseline for comparison in our empirical study). To tackle this challenge, our key motivation is to find some appropriate *confidence condition* such that it helps the online learner decide when we could skip the request of a label whenever the *confidence condition* is satisfied. To this end, we propose an idea to seek the confidence condition by estimating the difference between p_t and $\widehat{p}_t$. Intuitively, the smaller the difference, the more confident we have for the prediction made by the forecaster. Before introducing our proposed confidence conditions, for convenience of presentation, we introduce a notation: $\widehat{H}_{i,t} = \sum_{s=1}^{t}(1 - z_s)\ell(f_i(\mathbf{x}_s), y_s)$. It is easy to see $L_{i,t} = \widehat{L}_{i,t} + \widehat{H}_{i,t}$.

Active Exponentially Weighted Average Forecaster (AEWAF). We present a confidence condition for AEWAF in the following theorem, which guarantees a small difference between p_t and $\widehat{p}_t$.

Theorem 2. *For a small constant $\delta > 0$, $\max_{1 \le i,j \le N} |f_i(\mathbf{x}_t) - f_j(\mathbf{x}_t)| \le \delta$ implies $|p_t - \widehat{p}_t| \le \delta$.*

Proof. For the AEWAF strategy, the distance between p_t and $\widehat{p}_t$ is computed as follows:

$$\begin{aligned}
&|p_t - \widehat{p}_t| \\
&= \left| \frac{\sum_{i=1}^{N} \exp(-\eta L_{i,t-1}) f_i(\mathbf{x}_t)}{\sum_{i=1}^{N} \exp(-\eta L_{i,t-1})} - \frac{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1}) f_i(\mathbf{x}_t)}{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1})} \right| \\
&= \left| \frac{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1}) \exp(-\eta \widehat{H}_{i,t-1}) f_i(\mathbf{x}_t)}{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1}) \exp(-\eta \widehat{H}_{i,t-1})} \right. \\
&\qquad \left. - \frac{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1}) f_i(\mathbf{x}_t)}{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1})} \right| \\
&= \left| \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} \gamma_{i,j,t-1}(f_i(\mathbf{x}_t) - f_j(\mathbf{x}_t))}{\sum_{i=1}^{N} \sum_{j=1}^{N} \gamma_{i,j,t-1}} \right|
\end{aligned}$$

where

$$\gamma_{i,j,t-1} = \exp(-\eta \widehat{L}_{i,t-1}) \exp(-\eta \widehat{H}_{i,t-1}) \exp(-\eta \widehat{L}_{j,t-1}).$$

Thus, if $\max_{1 \le i,j \le N} |f_i(\mathbf{x}_t) - f_j(\mathbf{x}_t)| \le \delta$, it is easy to prove that $|p_t - \widehat{p}_t| \le \delta$. □

Active Greedy Forecaster (AGF). We now propose a confidence condition for AGF in the theorem below, which guarantees a small difference between $\widehat{p}_t$ and p_t.

Theorem 3. *For a small constant $\delta > 0$, $\max_{1 \le i \le N} |f_i(\mathbf{x}_t) - \bar{p}_t| \le \delta$ implies $|p_t - \widehat{p}_t| \le \delta$.*

Proof. We can bound p_t from the above as follows

$$\begin{aligned}
&p_t \\
&= \pi_{[0,1]} \left(\frac{1}{2} + \frac{1}{2\eta} \ln \frac{\sum_{i=1}^{N} \exp(-\eta L_{i,t-1} - \eta \ell(f_i(\mathbf{x}_t), 1))}{\sum_{i=1}^{N} \exp(-\eta L_{i,t-1} - \eta \ell(f_i(\mathbf{x}_t), 0))} \right) \\
&= \pi_{[0,1]} \Bigg(\frac{1}{2} + \\
&\qquad \frac{1}{2\eta} \ln \frac{\sum_{i=1}^{N} \exp(-\eta(\widehat{L}_{i,t-1} + \widehat{H}_{i,t-1}) - \eta \ell(f_i(\mathbf{x}_t), 1))}{\sum_{i=1}^{N} \exp(-\eta(\widehat{L}_{i,t-1} + \widehat{H}_{i,t-1}) - \eta \ell(f_i(\mathbf{x}_t), 0))} \Bigg) \\
&\le \pi_{[0,1]} \left(\frac{1}{2} + \frac{1}{2\eta} \ln \frac{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1} - \eta \ell(f_i(\mathbf{x}_t), 1))}{\sum_{i=1}^{N} \exp(-\eta \widehat{L}_{i,t-1} - \eta \ell(f_i(\mathbf{x}_t), 0))} \right) \\
&\qquad + \pi_{[0,1]} \left(\frac{1}{2\eta} \ln \frac{[\sum_{i=1}^{N} \alpha_{i,t} \exp(-\eta \widehat{H}_{i,t-1})]}{[\sum_{i=1}^{N} \beta_{i,t} \exp(-\eta \widehat{H}_{i,t-1})]} \right) \\
&= \widehat{p}_t + \pi_{[0,1]} \left(\frac{1}{2\eta} \ln \frac{[\sum_{i=1}^{N} \alpha_{i,t} \exp(-\eta \widehat{H}_{i,t-1})]}{[\sum_{i=1}^{N} \beta_{i,t} \exp(-\eta \widehat{H}_{i,t-1})]} \right)
\end{aligned}$$

where

$$\alpha_{i,t} = \frac{\exp\left(-\eta \left[\widehat{L}_{i,t-1} + \ell(f_i(\mathbf{x}_t), 1)\right]\right)}{\sum_{j=1}^{N} \exp\left(-\eta \left[\widehat{L}_{j,n-1} + \ell(f_j(\mathbf{x}_t), 1)\right]\right)},$$

$$\beta_{i,t} = \frac{\exp\left(-\eta \left[\widehat{L}_{i,t-1} + \ell(f_i(\mathbf{x}_t), 0)\right]\right)}{\sum_{j=1}^{N} \exp\left(-\eta \left[\widehat{L}_{j,n-1} + \ell(f_j(\mathbf{x}_t), 0)\right]\right)}, \; i \in [N].$$

Since $\forall i \in [N]$

$$\frac{\alpha_{i,t}}{\beta_{i,t}} = \frac{\sum_{j=1}^N \exp\left(-\eta\left[\widehat{L}_{j,t-1} + \ell(f_j(\mathbf{x}_t), 0)\right]\right)}{\sum_{j=1}^N \exp\left(-\eta\left[\widehat{L}_{j,t-1} + \ell(f_j(\mathbf{x}_t), 1)\right]\right)} \times$$
$$\frac{\exp\left(-\eta\left[\widehat{L}_{i,t-1} + \ell(f_i(\mathbf{x}_t), 1)\right]\right)}{\exp\left(-\eta\left[\widehat{L}_{i,t-1} + \ell(f_i(\mathbf{x}_t), 0)\right]\right)}$$
$$= \frac{\exp\left(-\eta\left[\ell(f_i(\mathbf{x}_t),1) - \ell(f_i(\mathbf{x}_t),0)\right]\right)}{\exp\left(\eta\left(2\bar{p}_t - 1\right)\right)}$$
$$= \frac{\exp\left(\eta\left(2f_i(\mathbf{x}_t) - 1\right)\right)}{\exp\left(\eta\left(2\bar{p}_t - 1\right)\right)}$$
$$= \exp\left(2\eta\left(f_i(\mathbf{x}_t) - \bar{p}_t\right)\right) \le \exp(2\eta\delta),$$

and $\ln x$ is an increasing function, we have

$$\ln \frac{\sum_{i=1}^N \alpha_{i,t} \exp(-\eta \widehat{H}_{i,t-1})}{\sum_{j=1}^N \beta_{j,t-1} \exp(-\eta \widehat{H}_{j,t-1})} \le 2\eta\delta$$

and $p_t \le \widehat{p}_t + \delta$. Similar to the above analysis, we have p_t lower bounded as $p_t \ge \widehat{p}_t - \delta$. □

Based on the above analysis of the confidence conditions, we can now present the general framework of active forecasters for online active learning with expert advice, which is summarized in Algorithm 1.

Algorithm 1 A Framework of Active Forecaster

```
Input: a pool of experts $f_i$, $i = 1, \dots, N$.
Initialize tolerance threshold $\delta$ and $\widehat{L}_{i,t} = 0$, $i \in [N]$.
for $t = 1, \dots, T$ do
    receive $\mathbf{x}_t$ and compute $f_i(\mathbf{x}_t)$, $i \in [N]$;
    compute $\bar{p}_t$ according to equation (5) and set $\widehat{p}_t = \pi_{[0,1]}(\bar{p}_t)$;
    if the confidence condition is satisfied then
        skip the label request for instance $\mathbf{x}_t$
    else
        request label $y_t$ and update $\widehat{L}_{i,t} = \widehat{L}_{i,t-1} + \ell(f_i(\mathbf{x}_t), y_t)$, $i \in [N]$;
    end if
end for
```

As shown in Algorithm 1, at each round, after receiving an input instance $\mathbf{x}_t$, we compute the prediction by each expert in the pool, i.e., $f_i(\mathbf{x}_t)$. Then, we examine if the confidence condition is satisfied. If so, we will skip the label request for this instance; otherwise, the learner will request the class label for this instance from the environment.

We now present a theorem about the upper bound of the regret of the two active forecasters, i.e.,

$$\widehat{R}_T = \widehat{L}_T - \min_{i \in [N]} L_{i,T}$$

where $\widehat{L}_T = \sum_{t=1}^T \ell(\widehat{p}_t, y_t)$, which is the overall loss experienced by the active forecaster.

Theorem 4. *Let the loss function $\ell(p, y) = |p - y|$, and denote by Q the total number of requested labels, i.e., $Q = \sum_{t=1}^T z_t$, then, for any T, $\eta > 0$ and for all $y_1, \dots, y_T \in \{0, 1\}$, the regret $\widehat{R}_T$ of the two Active Forecasters (AEWAF and AGF) can be bounded as:*

$$\widehat{R}_T = \widehat{L}_T - \min_{i \in [N]} L_{i,T} \le \frac{\ln N}{\eta} + \frac{\eta T}{8} + \delta(T - Q).$$

Proof. Firstly according to Theorem 2 and 3, we have the following bound for $\widehat{L}_T - L_T$:

$$\begin{aligned}\widehat{L}_T - L_T &= \sum_{t=1}^T (\ell(\widehat{p}_t, y_t) - \ell(p_t, y_t)) \\ &\le (T - Q) * |\widehat{p}_t - p_t| \le \delta(T - Q).\end{aligned}$$

Combining the above result with Theorem 1, we have the regret of the Active Forecasters bounded:

$$\widehat{R}_T = \widehat{L}_T - \min_{1 \le i \le N} L_{i,T} = (\widehat{L}_T - L_T) + (L_T - \min_{1 \le i \le N} L_{i,T})$$
$$\le \delta(T - Q) + \frac{\ln N}{\eta} + \frac{\eta T}{8}.$$

□

Remark. For the above theorem, if a learner requests the labels for every instance, i.e., Q=T, the bound reduces the bound of the regular forecasters. Based on the above theorem, we have the following corollary that shows the proposed Active Forecasters satisfy the Hannan consistency.

Corollary 5. *Consider $0 < a << T$, if we set $\eta = \sqrt{\frac{8 \ln N}{T(1+8a) - 8aQ}}$ and $\delta = a\eta$, then we have the regret achieved by the proposed algorithms bounded as $o(T)$.*

Proof. Following the result of Theorem 4, we have

$$\begin{aligned}\widehat{R}_T &\le \frac{\ln N}{\eta} + \frac{\eta T}{8} + \delta(T - Q) \\ &= \frac{\ln N}{\eta} + \eta\left((a + \frac{1}{8})T - Qa\right) \\ &= 2\sqrt{\ln N}\sqrt{T\left(a + \frac{1}{8}\right) - Qa}\end{aligned}$$

where the last equation holds under the condition $\frac{\ln N}{\eta} = \eta\left((a + \frac{1}{8})T - Qa\right)$, i.e., $\eta = \sqrt{\frac{8 \ln N}{T(1+8a) - 8aQ}}$, and as a result $\delta = a\sqrt{\frac{8 \ln N}{T(1+8a) - 8aQ}}$. Therefore, we have $\widehat{R}_T = o(T)$. □

4 Experimental Results

In this section, we evaluate the empirical performance of the proposed Active Forecasters for online active learning with expert advice tasks.

4.1 Experts and Compared Algorithms

To construct experts for an online sequential prediction task, we choose to build the pool of experts by adopting five well-known online learning algorithms [7, 5, 20], which include: (implemented as in [16])

- PERCEPTRON: the classical Perceptron algorithm [21];
- ROMMA: the Relaxed Online Maximum Margin Algorithm [18];
- ALMA$_p(\alpha)$: the Approximate Maximal Margin Algorithm [10];
- PA: the Passive-Aggressive online learning algorithm [4];
- AROW: the Adaptive Regularization Of Weights algorithm [6].

We compare the two proposed active learning algorithms (AEWAF and AGF) with the two regular forecasters (EWAF and GF) algorithm and their random variants as well, which are listed below:

- EWAF: the Exponentially Weighted Forecaster [2];
- GF: the Greedy Forecaster algorithm [2];
- REWAF: the Random Exponentially Weighted Forecaster, a variant of EWAF, which will randomly select the indices according to uniform distribution;
- RGF: the Random Greedy Forecaster algorithm, a variant of GF, which will randomly select the indices according to uniform distribution;
- AEWAF: the proposed Active Exponentially Weighted Forecaster algorithm;
- AGF: the proposed Active Greedy Forecaster algorithm.

4.2 Experimental Testbed and Setup

To evaluate the performance, we conduct experiments on a variety of benchmark datasets from web machine learning repositories. Table 1 shows the details of 9 datasets used in our experiments. All of them can be downloaded from LIBSVM website [1] and UCI machine learning repository [2]. These datasets are chosen fairly randomly in order to cover various aspects of datasets.

Table 1: Datasets used in the experiments.

Dataset	Name	# instances	# features
D1	a8a	32561	123
D2	codrna	271617	8
D3	covtype	581012	54
D4	gisette	7000	5000
D5	magic04	19020	10
D6	mushrooms	8124	112
D7	spambase	4601	57
D8	svmguide1	7089	4
D9	w8a	64700	300

All the expert algorithms learn a linear classifier for a binary classification task. The parameter p and α in ALMA$_p(\alpha)$ are set to be 2 and 0.9 respectively. The parameter C in PA is set to 5, and the parameter γ is set to 1 for AROW. To make fair comparisons, all the compared forecasters adopt the same setup. The learning rate η is set to $\sqrt{8 \ln N/T}$, for all the datasets and forecasters. The sampling ratio for requesting labeled data by the two random algorithms (REWAF and RGF) are set according to the ratio of labeled data requested by AEWAF and AGF using different δ values, respectively.

Each dataset is randomly divided into two subsets: a training set consisting of 20% of the entire data for training the experts; and a test set consisting of the remaining data for learning the forecasters. The five experts algorithms are applied on the training set to learn the five expert functions $\mathbf{u}_i \in \mathbb{R}^d,\ i \in [5]$, where d is the dimension of the instance. To satisfy the assumptions, we adopt $f_i(\mathbf{x}) = \pi_{[0,1]}(\mathbf{u}_i^\top \mathbf{x} + 0.5)$ as the expert functions. Then we test the forecasters on the test set. All the test experiments were conducted over 20 runs of different random permutations for each test set. All the results were reported by averaging over these 20 runs. For performance metric, we evaluate the forecasters by measuring the regret rate, ratio of requested labeled data, and the running time cost.

4.3 Evaluation of Regular Forecasters

Table 2 summarizes the average performance of the EWAF and GF algorithms for conventional online learning with expert advice on the benchmark datasets.

[1]`http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/`
[2]`http://www.ics.uci.edu/~mlearn/MLRepository.html`

Table 2: Evaluation of two regular forecasters (EWAF and GF) on all the datasets.

Dataset	Alg.	Measures	
		Regret (%)	Time (s)
D1	EWAF	0.286 ± 0.001	0.694
	GF	0.286 ± 0.001	0.841
D2	EWAF	0.207 ± 0.001	1.019
	GF	0.207 ± 0.001	1.282
D3	EWAF	0.066 ± 0.001	10.942
	GF	0.066 ± 0.001	13.428
D4	EWAF	0.609 ± 0.001	0.574
	GF	0.609 ± 0.001	0.593
D5	EWAF	0.373 ± 0.001	0.332
	GF	0.373 ± 0.001	0.414
D6	EWAF	0.483 ± 0.001	0.157
	GF	0.483 ± 0.001	0.194
D7	EWAF	0.758 ± 0.001	0.084
	GF	0.758 ± 0.001	0.103
D8	EWAF	0.506 ± 0.001	0.123
	GF	0.506 ± 0.001	0.154
D9	EWAF	0.164 ± 0.001	1.760
	GF	0.164 ± 0.001	2.031

Table 3: Evaluation of REWAF and AEWAF on all the dataset. R. denotes REWAF and A. denotes AEWAF. δ is set as 0.2.

Data	Alg.	Measures		
		Regret (%)	Query (%)	Time (s)
D1	R.	0.364 ± 0.017	77.35 ± 0.23	0.685
	A.	0.292 ± 0.001	77.44 ± 0.01	0.762
D2	R.	0.491 ± 0.025	42.48 ± 0.16	0.796
	A.	0.202 ± 0.001	42.46 ± 0.01	0.941
D3	R.	0.088 ± 0.002	74.26 ± 0.05	10.465
	A.	0.064 ± 0.000	74.25 ± 0.01	11.958
D4	R.	1.342 ± 0.102	45.40 ± 0.59	0.566
	A.	0.598 ± 0.003	45.40 ± 0.01	0.568
D5	R.	0.907 ± 0.076	40.94 ± 0.33	0.256
	A.	0.344 ± 0.004	41.00 ± 0.01	0.300
D6	R.	1.575 ± 0.067	10.80 ± 0.38	0.104
	A.	0.530 ± 0.004	10.63 ± 0.01	0.119
D7	R.	1.062 ± 0.066	71.64 ± 0.73	0.080
	A.	0.756 ± 0.004	71.67 ± 0.01	0.091
D8	R.	0.718 ± 0.047	49.36 ± 0.50	0.100
	A.	0.535 ± 0.006	49.41 ± 0.01	0.117
D9	R.	0.248 ± 0.006	40.79 ± 0.17	1.489
	A.	0.169 ± 0.000	40.82 ± 0.01	1.636

From the experimental results in Table 2, we can draw several observations. First, the regret rates of EWAF and GF on every dataset are almost the same, which is consistent to the theoretical result that shows that they share the same regret bound. Second, EWAF consumes slightly less time cost than GF for all the cases due to the difference of their solutions. Finally, we found that the larger the dataset size, the smaller the average regret value achieved by the two algorithms. This is consistent with the Hannan property satisfied by the two algorithms, i.e., the average regret is negligible when T is very large.

4.4 Evaluation of Active Forecasters on Fixed Ratio of Queries

In this subsection, the tolerance threshold δ for AEWAF and AGF is set as 0.2. Table 3 summarizes the average performance of the REWAF and AEWAF algorithms over the experimental datasets. From the experimental results, we can draw several observations as follows.

First of all, since we choose the sampling threshold ρ according to the ratios of required labels by AEWAF using a fixed tolerances δ, the differences between the ratios of requested labeled data for AEWAF and REWAF are not statistically significant, which has been verified by statistical t-tests. This implies that the statistical differences between the regret rates achieved by AEWAF and REWAF, if any, are not caused by the differences between their ratios of the requested labeled data.

Second, compared with REWAF, AEWAF achieves statistically lower regret rates on all the datasets, which validates the effectiveness of the proposed active learning strategy and also indicates the importance of exploiting the degree of agreements between different experts. In addition, AEWAF can achieve comparable regret rates with EWAF by requesting a significantly less amount of labels; while REWAF suffers significantly more regret rates by requesting the same amount of labels. This shows that AEWAF could be an attractive alternative to the EWAF in order to save the expensive labeling efforts in a real application.

Third, the time cost of the AEWAF algorithm is in general comparable to or slightly higher than that of the REWAF algorithm because the proposed confidence conditions can be evaluated rather efficiently.

Finally, we would also like to examine if the proposed active learning strategy can be generalized to different types of forecasting algorithms. To this purpose, we also evaluate the performance of the RGF and AGF algorithms. Table 4 summarizes the experimental results on all the datasets. As compared to the last experiment, similar observations can be drawn from the experimental results. We found that AEWAF and AGF request almost the same ratios of labels and achieve comparable regret rates on all the datasets; while REWAF and RGF achieve comparable regret rates, which are significantly higher than those of the two proposed active algorithms. These results indicate that the proposed active learning strategy can be generalized to different types of forecasting algorithms, and again validate the efficacy of the proposed active learning algorithms.

Table 4: Evaluation of RGF and AGF on all the dataset. R. denotes RGF and A. denotes AGF. δ is set as 0.2.

Data	Alg.	Measures		
		Regret (%)	Query (%)	Time (s)
D1	R.	0.362 ± 0.022	76.61 ± 0.26	0.905
	A.	0.288 ± 0.003	76.56 ± 0.04	0.974
D2	R.	0.474 ± 0.028	42.38 ± 0.21	1.363
	A.	0.199 ± 0.002	42.40 ± 0.01	1.542
D3	R.	0.089 ± 0.002	74.21 ± 0.06	14.416
	A.	0.063 ± 0.001	74.21 ± 0.01	15.959
D4	R.	1.344 ± 0.088	45.53 ± 0.71	0.610
	A.	0.592 ± 0.005	45.27 ± 0.04	0.611
D5	R.	0.923 ± 0.096	40.13 ± 0.39	0.442
	A.	0.335 ± 0.009	40.32 ± 0.07	0.498
D6	R.	1.633 ± 0.053	10.21 ± 0.41	0.206
	A.	0.543 ± 0.005	10.22 ± 0.07	0.231
D7	R.	1.088 ± 0.080	71.18 ± 0.82	0.113
	A.	0.752 ± 0.007	71.27 ± 0.06	0.126
D8	R.	0.726 ± 0.043	46.97 ± 0.48	0.164
	A.	0.559 ± 0.015	47.01 ± 0.40	0.185
D9	R.	0.247 ± 0.005	40.55 ± 0.21	2.110
	A.	0.171 ± 0.001	40.59 ± 0.01	2.303

4.5 Evaluation of Active Forecasters on Varied Ratios of Queries

Firstly, Figure (1) shows the performance of the REWAF and AEWAF algorithms on mushrooms with varied ratios of queries. AEWAF outperforms REWAF with all the ratios of queries, which verifies the proposed active strategies are effective and promising.

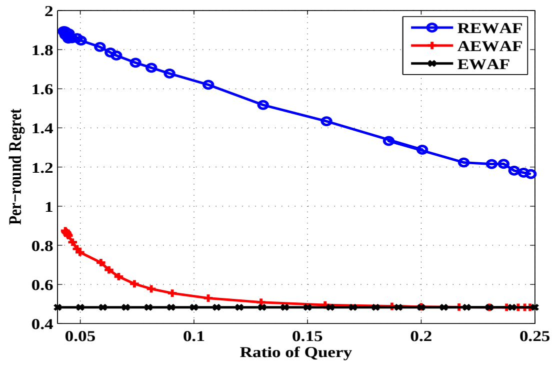


Figure 1: Comparison of REWAF and AEWAF on mushrooms.

Secondly, Figure (2) shows the performance of the RGF and AGF algorithms on mushrooms with varied ratios of queries. AEWAF outperforms REWAF with all the ratios of queries, which again verifies the proposed active strategies are effective and promising.

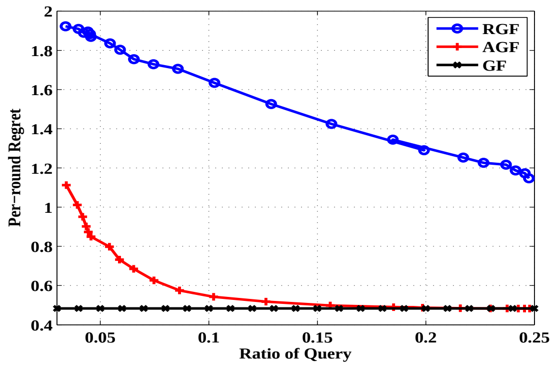


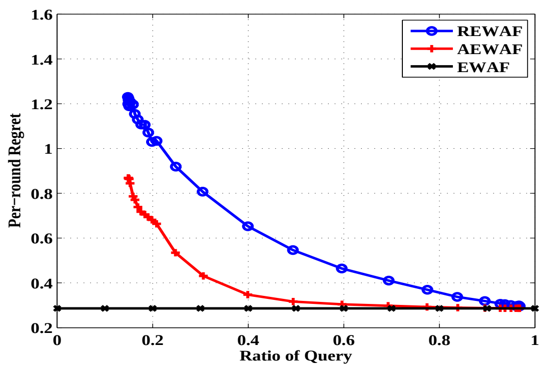
Figure 2: Comparison of RGF and AGF on mushrooms.

Finally, Figure (3) and (4) shows the comparisons of REWAF and AEWAF, RGF and AGF on all the remaining datasets, respectively, where similar observations can be found from the results.

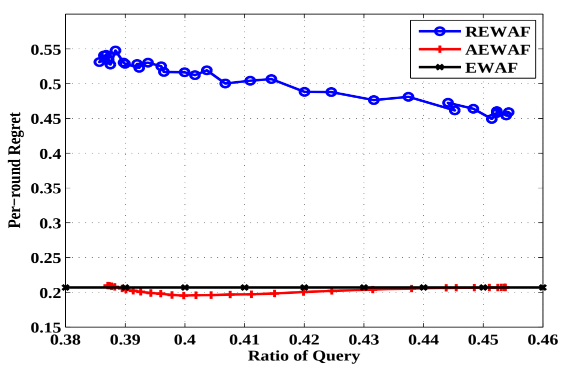
5 Conclusion

This paper addressed a new problem of active learning with expert advice for online sequential prediction tasks. We proposed two novel strategies for active learning with expert advice by extending two existing forecasting algorithms in an online active learning setting. We analyze the theoretical regret bounds for the proposed algorithms, which guarantee the proposed algorithms satisfy the important Hannan consistency. We have conducted an extensive set of experiments to evaluate the efficacy of the algorithms. Promising empirical results validate the effectiveness of our technique.

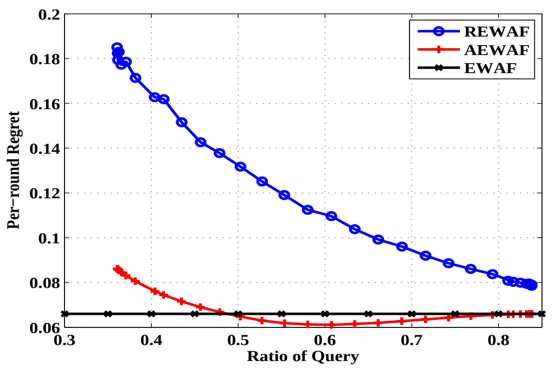
Despite the encouraging results, some limitations and open challenges of the current work remain. One issue is about the settings of the learning rate η and tolerance parameter δ, which were fixed manually in our experiments. It would be more attractive if one is able to design a self-tuned strategy for the active learning task. Further, the current regret bounds may be further improved, e.g., by adopting different loss functions or other strategies. Another future work may be exploring the principles of semi-supervised learning for improving active learning with expert advice [15]. These issues can be further explored in the future work.



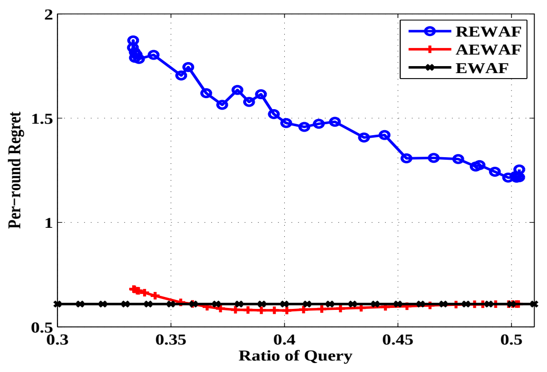
(a) a8a



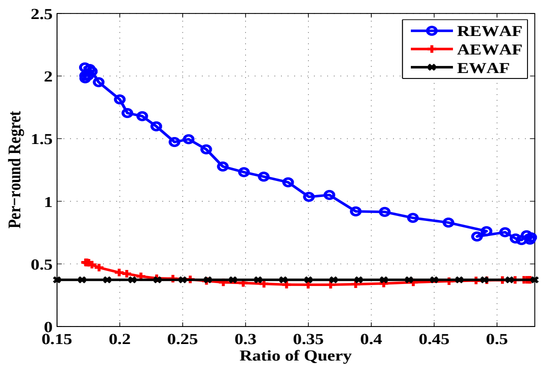
(b) codrna



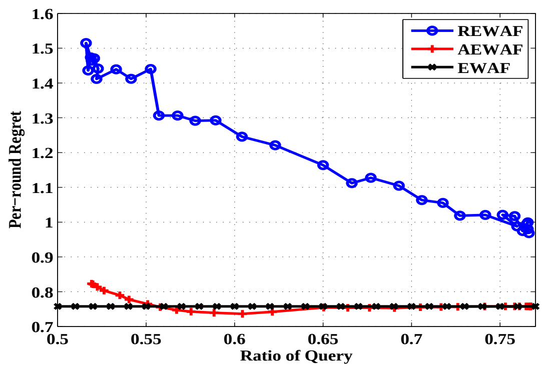
(c) covtype



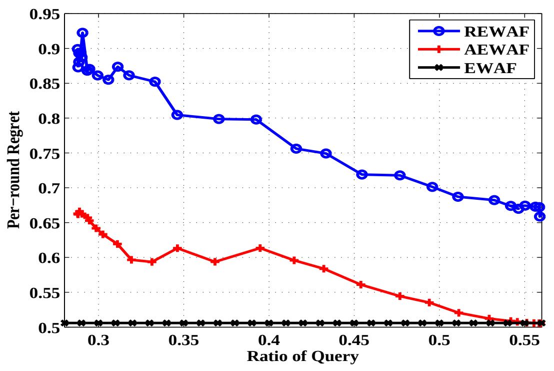
(d) gisette



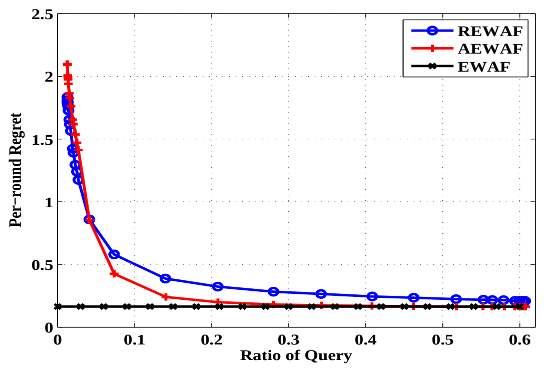
(e) magic04



(f) spambase

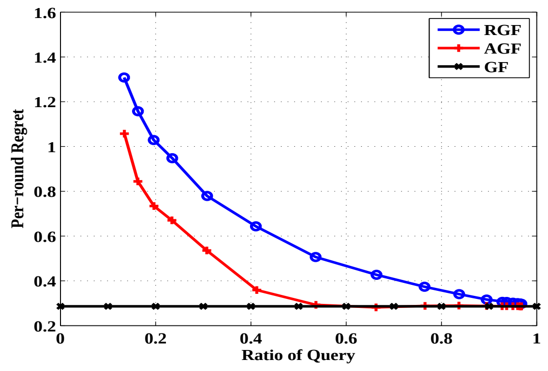


(g) svmguide1

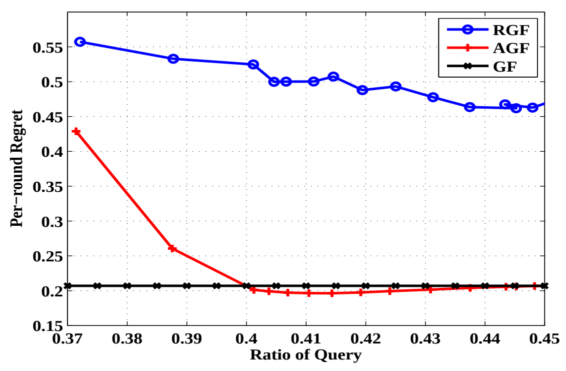


(h) w8a

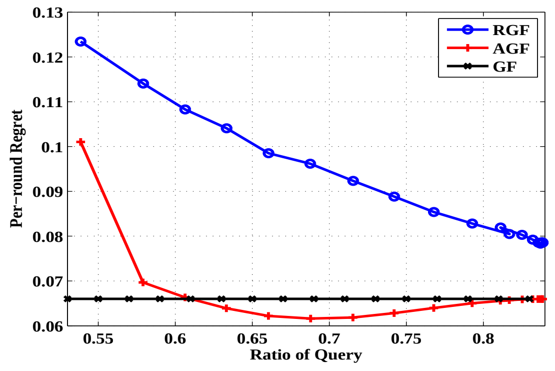
Figure 3: Comparison of regret rates with respect to varied ratios of queries.



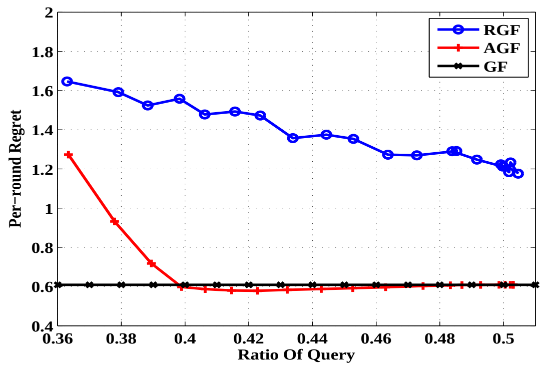
(a) a8a



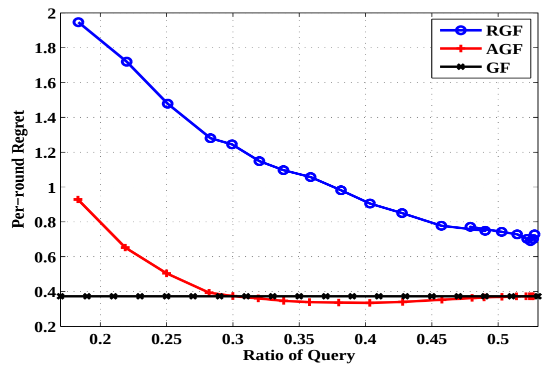
(b) codrna



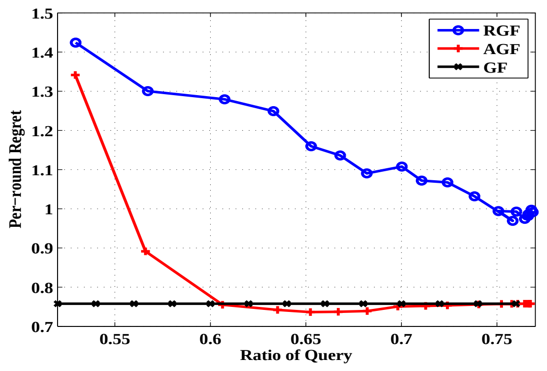
(c) covtype



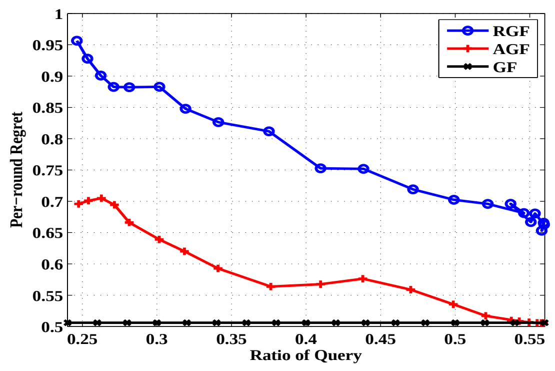
(d) gisette



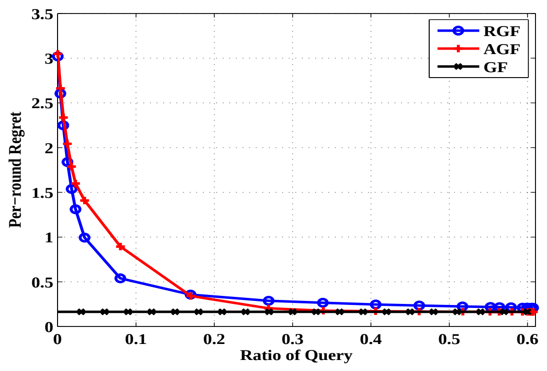
(e) magic04



(f) spambase



(g) svmguide1



(h) w8a

Figure 4: Comparison of regret rates with respect to varied ratios of queries.

References

[1] Olivier Bousquet and Manfred K. Warmuth. Tracking a small set of experts by mixing past posteriors. *Journal of Machine Learning Research*, 3:363–396, 2002.

[2] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[3] Nicolò Cesa-Bianchi, Yoav Freund, David Haussler, David P. Helmbold, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. *J. ACM*, 44(3):427–485, 1997.

[4] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[5] Koby Crammer, Mark Dredze, and Fernando Pereira. Exact convex confidence-weighted learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 345–352, 2008.

[6] Koby Crammer, Alex Kulesza, and Mark Dredze. Adaptive regularization of weight vectors. In *NIPS*, pages 414–422, 2009.

[7] Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *Proceedings of the 25th International Conference on Machine Learning (ICML2008)*, pages 264–271, 2008.

[8] Dean P. Foster and Rakesh V. Vohra. A randomization rule for selecting forecasts. *Oper. Res.*, 41:704–709, July 1993.

[9] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55:119–139, August 1997.

[10] Claudio Gentile. A new approximate maximal margin classification algorithm. *Journal of Machine Learning Research*, 2:213–242, 2001.

[11] J. Hannan. Approximation to bayes risk in repeated plays. *Contributions to the Theory of Games*, 3:97–139, 1957.

[12] David Haussler, Jyrki Kivinen, and Manfred K. Warmuth. Tight worst-case loss bounds for predicting with expert advice. In *EuroCOLT*, pages 69–83, 1995.

[13] Mark Herbster and Manfred K. Warmuth. Tracking the best expert. *Machine Learning*, 32(2):151–178, 1998.

[14] Steven C. H. Hoi, Rong Jin, Peilin Zhao, and Tianbao Yang. Online multiple kernel classification. *Machine Learning*, 90(2):289–316, 2013.

[15] Steven C.H. Hoi, Rong Jin, Jianke Zhu, and Michael R Lyu. Semisupervised svm batch mode active learning with applications to image retrieval. *ACM Transactions on Information Systems (TOIS)*, 27(3):16, 2009.

[16] Steven C.H. Hoi, Jialei Wang, and Peilin Zhao. *LIBOL: A Library for Online Learning Algorithms*. Nanyang Technological University, 2012.

[17] Guangxia Li, Steven C. H. Hoi, Kuiyu Chang, and Ramesh Jain. Micro-blogging sentiment detection by collaborative online learning. In *ICDM*, pages 893–898, 2010.

[18] Yi Li and Philip M. Long. The relaxed online maximum margin algorithm. In *Advances in Neural Information Processing Systems (NIPS)*, pages 498–504, 1999.

[19] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2):212–261, 1994.

[20] Francesco Orabona and Koby Crammer. New adaptive algorithms for online classification. In *NIPS*, pages 1840–1848, 2010.

[21] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

[22] Volodimir G. Vovk. Aggregating strategies. In *Proceedings of the third annual workshop on Computational learning theory (COLT'90)*, pages 371–386, 1990.

[23] Jialei Wang, Peilin Zhao, and Steven C. H. Hoi. Cost-sensitive online classification. In *ICDM*, pages 1140–1145, 2012.

[24] Jialei Wang, Peilin Zhao, and Steven C. H. Hoi. Exact soft confidence-weighted learning. In *ICML*, 2012.

[25] Jialei Wang, Peilin Zhao, Steven C.H. Hoi, and Rong Jin. Online feature selection and its applications. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–14, 2013.

[26] Peilin Zhao, Steven C. H. Hoi, and Rong Jin. Double updating online learning. *Journal of Machine Learning Research*, 12:1587–1615, 2011.

Bennett-type Generalization Bounds: Large-deviation Case and Faster Rate of Convergence

Chao Zhang
The Biodesign Institute
Arizona State University
Tempe, AZ 85287, USA

Abstract

In this paper, we present the Bennett-type generalization bounds of the learning process for i.i.d. samples, and then show that the generalization bounds have a faster rate of convergence than the traditional results. In particular, we first develop two types of Bennett-type deviation inequality for the i.i.d. learning process: one provides the generalization bounds based on the uniform entropy number; the other leads to the bounds based on the Rademacher complexity. We then adopt a new method to obtain the alternative expressions of the Bennett-type generalization bounds, which imply that the bounds have a faster rate $o(N^{-\frac{1}{2}})$ of convergence than the traditional results $O(N^{-\frac{1}{2}})$. Additionally, we find that the rate of the bounds will become faster in the large-deviation case, which refers to a situation where the empirical risk is far away from (at least not close to) the expected risk. Finally, we analyze the asymptotical convergence of the learning process and compare our analysis with the existing results.

1 Introduction

In learning theory, one of the major concerns is to obtain the generalization bounds of the learning process for i.i.d. samples, which measure the probability that a function obtained in the i.i.d. learning process has a sufficiently small error [Vapnik, 1998, Bousquet et al., 2004]. Generalization bounds have been widely used to study many topics of learning theory, *e.g.*, the consistency of ERM-based learning processes [Vapnik, 1998], the asymptotic convergence of empirical process [Van der Vaart and Wellner, 1996] and the learnability of learning models [Blumer et al., 1989].

Deviation (or concentration) inequalities play an essential role in obtaining generalization bounds. There are many popular deviation and concentration inequalities, *e.g.*, Hoeffding's inequality, McDiarmid's inequality, Bennett's inequality, Bernstein's inequality and Talagrand's inequality. Among them, Hoeffding's inequality and McDiarmid's inequality have been intensively used to obtain the generalization bounds based on the covering number (or uniform entropy number) and the Rademacher complexity, respectively [Mendelson, 2003, Van der Vaart and Wellner, 1996]. To obtain the generalization bounds based on the VC dimension, Vapnik [1998] applied some classical inequalities, for example, Chernoff's inequality and Hoeffding's inequality, but also developed specific concentration inequalities. Bartlett et al. [2005] presented generalization bounds based on the local Rademacher complexity by using Talagrand's inequality. Additionally, there are also other works to study the generalization bound in statistical learning theory [Ben-David et al., 2010, Mohri and Rostamizadeh, 2010]. However, to our best knowledge, there is little theoretical investigation into the generalization bounds derived from Bennett's inequality.

1.1 Motivation

The paper is motivated by the difference between Hoeffding's inequality [Hoeffding, 1963] and Bennett's inequality [Bennett, 1962]. It is well-known that Hoeffding's inequality is achieved by only exploiting the expectation information, and Bennett's inequality is based on both of expectation and variance. Intuitively, the Bennett-type generalization bounds should have a faster rate of convergence than that of the Hoeffding-type results. However, to our best knowledge, this issue has not been well explored in the literature. In this paper, we will give a theoretical argument to show the faster rate of the Bennett-type results.

1.2 Overview of Main Results

In this paper, we are mainly concerned with the following three aspects: (1) Bennett-type deviation inequality; (2) Bennett-type generalization bounds; (3) rate of convergence.

By extending the classical Bennett's inequality, we use the martingale method to develop a Bennett-type deviation inequality for the case of multiple random variables and this inequality leads to the generalization bounds based on the uniform entropy number (UEN). Moreover, under the bounded-difference condition, we present another Bennett-type deviation inequality that is similar to McDiarmid's inequality. We use this deviation inequality to further obtain the generalization bounds based on the Rademacher complexity. Note that the aforementioned generalization bounds are said to be of Bennett-type, because they are derived from the Bennett-type deviation inequalities.

In order to analyze the rate of convergence of the bounds, one needs to obtain the alternative expressions of the Bennett-type generalization bounds. Differing from the Hoeffding-type bounds [see (7)], the expression of Bennett-type bounds [see (16)] are not well-defined and thus it is difficult to directly present the alternative expressions of the Bennett-type bounds.[1] Instead, one generally transforms the Bennett-type bound to the Bernstain-type one and then obtains the corresponding alternative expression showing that the bound has the rate $O(N^{-\frac{1}{2}})$ of convergence, which is in accordance with the classical result (8). Here, we use a new method to obtain another alternative expression of the Bennett-type bound and show that the Bennett-type bounds have a faster rate $o(N^{-\frac{1}{2}})$ of convergence than the classical result $O(N^{-\frac{1}{2}})$. Additionally, we point out that the rate of the bounds will become faster as the discrepancy between the empirical risk and the expected risk becomes larger. This situation is called as the large-deviation case [see Remark 4.3].

Note that it is well-known that the rate of the empirical Rademacher complexity is up to $O(N^{-\frac{1}{2}})$ [see (10)], and thus it seems to be a contradiction that the Bennett-type bounds have a faster rate $o(N^{-\frac{1}{2}})$) [see Remark 5.1]. An explanation to the contradiction is given to support the theoretical findings on the faster rate of convergence.

[1]The reason is that it is difficult to directly obtain the analytical expression of the inverse function of $\Gamma(x) = x - (x+1)\ln(x+1)$.

1.3 Organization of the Paper

The rest of this paper is organized as follows. Section 2 formalizes the main issues of this paper and briefs the classical results. In Section 3, we present two Bennett-type deviation inequalities for the case of multiple random variables. Section 4 provides the generalization bounds and a new upper bound of the Rademacher complexity is presented in Section 5. In Section 6, we analyze the asymptotic convergence of the i.i.d. learning process and the last section concludes the paper. The proofs of the main results are given in the long version of this paper.[2]

2 Preliminaries

In this section, we first introduce some notations to formalize the proposed research of this paper, and then present the definitions of the covering number, the uniform entropy number and the Rademacher complexity. Moreover, we also brief the classical results of the Heoffding-type generalization bounds of the i.i.d. learning process.

2.1 Problem Setup

Let $\mathcal{X} \subset \mathbb{R}^I$ and $\mathcal{Y} \subset \mathbb{R}^J$ be an input space and its corresponding output space, respectively. Denote $\mathcal{Z} := \mathcal{X} \times \mathcal{Y} \subset \mathbb{R}^I \times \mathbb{R}^J$ with $K = I + J$. Let $\mathcal{G} \subset \mathcal{Y}^{\mathcal{X}}$ be a function class with the domain $\mathcal{X}$ and the range $\mathcal{Y}$. Given a loss function $\ell : \mathcal{Y}^2 \to \mathbb{R}$, it is expected to find a function $g^* \in \mathcal{G} : \mathcal{X} \to \mathcal{Y}$ that minimizes the expected risk over $\mathcal{G}$

$$\mathrm{E}(\ell \circ g) := \int \ell(g(\mathbf{x}), \mathbf{y}) d\mathrm{P}(\mathbf{z}), \quad g \in \mathcal{G}, \tag{1}$$

where $\mathrm{P}(\mathbf{z})$ stands for the distribution of $\mathbf{z} = (\mathbf{x}, \mathbf{y})$.

Generally, the distribution $\mathrm{P}(\mathbf{z})$ is unknown and thus the target function g^* cannot be directly obtained by minimizing (1). Instead, we can apply the empirical risk minimization (ERM) principle to handle this issue [Vapnik, 1998]. Given a function class $\mathcal{G}$ and a set of i.i.d. samples $\mathbf{Z}_1^N := \{\mathbf{z}_n\}_{n=1}^N$ drawn from $\mathcal{Z}$, we define the empirical risk of $g \in \mathcal{G}$ as

$$\mathrm{E}_N(\ell \circ g) := \frac{1}{N} \sum_{n=1}^{N} \ell(g(\mathbf{x}_n), \mathbf{y}_n), \tag{2}$$

which is considered as an approximation to the expected risk (1). Let $g_N \in \mathcal{G}$ be the function that minimizes the empirical risk (2) over $\mathcal{G}$ and deem g_N as an estimate to g^* with respect to $\mathbf{Z}_1^N$.

In the aforementioned i.i.d. learning process, we are mainly interested in the following two types of quantities:

[2]https://sites.google.com/site/czhang1015/

- $\mathrm{E}(\ell \circ g_N) - \mathrm{E}_N(\ell \circ g_N)$, which corresponds to the estimation of the expected risk from an empirical quantity;
- $\mathrm{E}(\ell \circ g_N) - \mathrm{E}(\ell \circ g^*)$, which corresponds to the performance of the ERM-based algorithm.

Recalling (1) and (2), since

$$\mathrm{E}_N(\ell \circ g^*) - \mathrm{E}_N(\ell \circ g_N) \geq 0,$$

we have

$$\begin{aligned}\mathrm{E}(\ell \circ g_N) =& \mathrm{E}(\ell \circ g_N) - \mathrm{E}(\ell \circ g^*) + \mathrm{E}(\ell \circ g^*)\\ \leq& \mathrm{E}_N(\ell \circ g^*) - \mathrm{E}_N(\ell \circ g_N) + \mathrm{E}(\ell \circ g_N)\\ &- \mathrm{E}(\ell \circ g^*) + \mathrm{E}(\ell \circ g^*)\\ \leq& 2\sup_{g\in\mathcal{G}} \big|\mathrm{E}(\ell \circ g) - \mathrm{E}_N(\ell \circ g)\big| + \mathrm{E}(\ell \circ g^*),\end{aligned}$$

and thus

$$0 \leq \mathrm{E}(\ell \circ g_N) - \mathrm{E}(\ell \circ g^*) \leq 2\sup_{g\in\mathcal{G}} \big|\mathrm{E}(\ell \circ g) - \mathrm{E}_N(\ell \circ g)\big|,$$

with

$$\mathrm{E}(\ell \circ g_N) - \mathrm{E}_N(\ell \circ g_N) \leq \sup_{g\in\mathcal{G}} \big|\mathrm{E}(\ell \circ g) - \mathrm{E}_N(\ell \circ g)\big|.$$

This shows that the asymptotic behaviors of the aforementioned two quantities, when the sample number N goes to *infinity*, can both be described by the supremum

$$\sup_{g\in\mathcal{G}} \big|\mathrm{E}(\ell \circ g) - \mathrm{E}_N(\ell \circ g)\big|, \tag{3}$$

which is the so-called generalization bound of the i.i.d. learning process.

For convenience, we define the loss function class

$$\mathcal{F} := \{\mathbf{z} \mapsto \ell(g(\mathbf{x}), \mathbf{y}) : g \in \mathcal{G}\},$$

and call $\mathcal{F}$ as the function class in the rest of this paper. By (1) and (2), given a sample set $\mathbf{Z}_1^N$ drawn from $\mathcal{Z}$, we briefly denote for any $f \in \mathcal{F}$,

$$\mathrm{E}f := \int f(\mathbf{z})d\mathrm{P}(\mathbf{z});\ \mathrm{E}_N f := \frac{1}{N}\sum_{n=1}^{N} f(\mathbf{z}_n).$$

Thus, we rewrite the generalization bound (3) as

$$\sup_{f\in\mathcal{F}} \big|\mathrm{E}f - \mathrm{E}_N f\big|.$$

2.2 Complexity Measures of Function Classes

Generally, the generalization bound of a certain learning process is achieved by incorporating the complexity measure of the function class, *e.g.*, the covering number, the VC dimensions and the Rademacher complexity. This paper is mainly concerned with the covering number, the uniform entropy number (UEN) and the Rademacher complexity.

2.2.1 Covering Number and Uniform Entropy Number (UEN)

The following is the definition of the covering number and we refer to Mendelson [2003] for details.

Definition 2.1 *Let $\mathcal{F}$ be a function class and d be a metric on $\mathcal{F}$. For any $\xi > 0$, the covering number of $\mathcal{F}$ at radius ξ with respect to the metric d, denoted by $\mathcal{N}(\mathcal{F}, \xi, d)$ is the minimum size of a cover of radius ξ.*

For clarity of presentation, we give a useful notation for the following discussion. Given a sample set $\mathbf{Z}_1^N := \{\mathbf{z}_n\}_{n=1}^N$ drawn from $\mathcal{Z}$, we denote $\mathbf{Z'}_1^N := \{\mathbf{z'}_n\}_{n=1}^N$ as the ghost-sample set drawn from $\mathcal{Z}$ such that the ghost sample $\mathbf{z'}_n$ has the same distribution as $\mathbf{z}_n$ for any $1 \leq n \leq N$. Denote $\mathbf{Z}_1^{2N} := \{\mathbf{Z}_1^N, \mathbf{Z'}_1^N\}$. Setting the metric d as the $\ell_p(\mathbf{Z}_1^{2N})$ $(p > 0)$ norm, we then obtain the covering number $\mathcal{N}\left(\mathcal{F}, \xi, \ell_p(\mathbf{Z}_1^{2N})\right)$.

The uniform entropy number (UEN) is a variant of the covering number and we refer to Mendelson [2003] for details as well. By setting the metric $\ell_p(\mathbf{Z}_1^N)$ $(p > 0)$, the UEN is defined as follows:

$$\ln \mathcal{N}_p(\mathcal{F}, \xi, N) := \sup_{\mathbf{Z}_1^{2N}} \ln \mathcal{N}\left(\mathcal{F}, \xi, \ell_p(\mathbf{Z}_1^N)\right). \tag{4}$$

2.2.2 Rademacher Complexity

The Rademacher complexity is one of the most frequently used complexity measures of function classes and we refer to Bousquet et al. [2004] for details.

Definition 2.2 *Let $\mathcal{F}$ be a function class and $\{\mathbf{z}_n\}_{n=1}^N$ be a sample set drawn from $\mathcal{Z}$. Denote $\{\sigma_n\}_{n=1}^N$ as a set of random variables independently taking either value of $\{-1, 1\}$ with equal probability. The Rademacher complexity of $\mathcal{F}$ is defined as*

$$\mathcal{R}(\mathcal{F}) := \mathrm{E}\sup_{f\in\mathcal{F}} \left\{\frac{1}{N}\sum_{n=1}^{N} \sigma_n f(\mathbf{z}_n)\right\} \tag{5}$$

with its empirical version

$$\mathcal{R}_N(\mathcal{F}) := \mathrm{E}_\sigma \sup_{f\in\mathcal{F}} \left\{\frac{1}{N}\sum_{n=1}^{N} \sigma_n f(\mathbf{z}_n)\right\}, \tag{6}$$

where E *stands for the expectation taken with respect to all random variables $\{\mathbf{z}_n\}_{n=1}^N$ and $\{\sigma_n\}_{n=1}^N$, and* E_σ *stands for the expectation only taken with respect to random variables $\{\sigma_n\}_{n=1}^N$.*

2.3 Classical Hoeffding-type Generalization Bounds

Next, we summarize some classical results of the generalization bounds. Note that the following bounds are said to be of Hoeffding-type, because they are derived from (or strongly related to) Hoeffding's inequality and Hoeffding's lemma [Hoeffding, 1963].

First, based on the covering number, one can use Hoeffding's inequality (or Hoeffding's lemma) to obtain the following generalization bound w.r.t. a function class $\mathcal{F}$ with the range $[a, b]$ [see Mendelson, 2003, Theorem 2.3]: for any $N \geq \frac{8(b-a)^2}{\xi^2}$,

$$\begin{aligned}&\Pr\left\{\sup_{f\in\mathcal{F}}\left|\mathrm{E}f - \mathrm{E}_N f\right| > \xi\right\}\\ \leq& 8\mathrm{E}\mathcal{N}\left(\mathcal{F}, \xi/8, \ell_1(\mathbf{Z}_1^{2N})\right)\exp\left\{-\frac{N\xi^2}{32(b-a)^2}\right\}, \quad (7)\end{aligned}$$

which is one of the most frequently used generalization results in statistical learning theory. Because of its well-defined expression, we can directly obtain its alternative expression: for any $N \geq \frac{8(b-a)^2}{\xi^2}$, with probability at least $1-\epsilon$,

$$\begin{aligned}&\sup_{f\in\mathcal{F}}\left|\mathrm{E}_N f - \mathrm{E}f\right| \quad (8)\\ \leq& O\left(\left(\frac{\ln \mathrm{E}\mathcal{N}\left(\mathcal{F}, \xi/8, \ell_1(\mathbf{Z}_1^{2N})\right) - \ln(\epsilon/8)}{N}\right)^{\frac{1}{2}}\right).\end{aligned}$$

Following the alternative expression (8), it is observed that the generalization bound $\sup_{f\in\mathcal{F}}\left|\mathrm{E}_N f - \mathrm{E}f\right|$ has a convergence rate of $O(N^{-\frac{1}{2}})$.

On the other hand, McDiarmid's inequality can provide the following generalization bound based on the Rademacher complexity [see Bousquet et al., 2004, Theorem 5]: for any $\epsilon > 0$ and $f \in \mathcal{F}$, with probability at least $1-\epsilon$,

$$\begin{aligned}\mathrm{E}f \leq& \mathrm{E}_N f + 2\mathcal{R}(\mathcal{F}) + (b-a)\left(\frac{\ln(1/\epsilon)}{N}\right)^{\frac{1}{2}}\\ \leq& \mathrm{E}_N f + 2\mathcal{R}_N(\mathcal{F}) + 3(b-a)\left(\frac{\ln(2/\epsilon)}{2N}\right)^{\frac{1}{2}}, \quad (9)\end{aligned}$$

which is also of Hoeffding-type, because McDiarmid's inequality is actually derived from Hoeffiding's lemma under the condition that f has the bounded-differences property [see Bousquet et al., 2004, Theorem 6]. Furthermore, it is followed from Sudakov minoration for Rademacher processes[3] [Talagrand, 1994b,a, Latała, 1997] and Massart's finite class lemma (Dudley's entropy integral)[4] [Mendelson, 2003, Van der Vaart and Wellner, 1996] that:

$$\begin{aligned}&\frac{c}{\ln N}\sup_{\alpha>0}\alpha\sqrt{\frac{\ln\mathcal{N}(\mathcal{F}, \xi, \ell_2(\mathbf{Z}_1^N))}{N}} \leq \mathcal{R}_N(\mathcal{F}) \quad (10)\\ \leq& \inf_{\epsilon>0}\left\{4\epsilon + 12\int_{\epsilon}^{\infty}\sqrt{\frac{\ln\mathcal{N}(\mathcal{F}, \xi, \ell_2(\mathbf{Z}_1^N))}{N}}d\xi\right\},\end{aligned}$$

which shows the lower and the upper bounds of the empirical Rademacher complexity $\mathcal{R}_N(\mathcal{F})$. The combination of (9) and (10) implies that the rate of convergence of $\sup_{f\in\mathcal{F}}\left|\mathrm{E}_N f - \mathrm{E}f\right|$ is up to $O(N^{-\frac{1}{2}})$ as well.

3 Bennett-type Deviation Inequalities

By extending the classical Bennett's inequality [Bennett, 1962] to the case of multiple random variables, this section will present two Bennett-type deviation inequalities for the i.i.d. learning process. We also refer to Bousquet [2002] for the application of Bennett's inequality in the empirical process.

Theorem 3.1 *Let f be a function with the range $[a, b]$ and $\mathbf{Z}_1^N = \{\mathbf{z}_n\}_{n=1}^N$ be a set of i.i.d. samples drawn from $\mathcal{Z}$. Define a function $F : \mathbb{R}^{KN} \to \mathbb{R}$ as*

$$F\left(\mathbf{Z}_1^N\right) := \sum_{n=1}^{N} f(\mathbf{z}_n). \quad (11)$$

Then, we have for any $0 < \xi < N(b-a)$,

$$\Pr\left\{\left|\mathrm{E}\{F\} - F(\mathbf{Z}_1^N)\right| > \xi\right\} \leq 2\mathrm{e}^{N\Gamma\left(\frac{\xi}{N(b-a)}\right)}, \quad (12)$$

where

$$\Gamma(x) := x - (1+x)\ln(1+x). \quad (13)$$

The proof of this result is processed by the martingale method. Compared to the classical Bennett's inequality, this result is valid for the case of multiple random variables and provides the convenience to obtain the generalization bound of the i.i.d. learning process. Especially, the two inequalities will coincide, if there is only one random variable.

Moreover, recalling the classical McDiarmid's inequality [see Bousquet et al., 2004, Theorem 6], it is actually derived from Hoeffiding's lemma under the condition that f has the bounded-differences property. Thus, McDiarmid's inequality has a similar expression to that of Hoeffding's inequality and the two inequalities coincide if there is only one random variable. Similarly, we obtain another Bennett-type deviation inequality under the bounded-difference condition:

Theorem 3.2 *Let $\mathbf{z}_1, \cdots, \mathbf{z}_N$ be N independent random variables taking values from $\mathcal{Z}$. Assume that*

[3]See http://www.cs.berkeley.edu/ bartlett/courses/281b-sp08/19.pdf

[4]See http://ttic.uchicago.edu/~karthik/dudley.pdf

there exists a positive constant such that the function $H : \mathcal{Z}^N \to \mathbb{R}$ satisfies the bounded-difference condition: for any $1 \leq n \leq N$,

$$\sup_{\mathbf{z}_1,\cdots,\mathbf{z}_N,\mathbf{z}'_n} \Big| H\big(\mathbf{z}_1,\cdots,\mathbf{z}_n,\cdots,\mathbf{z}_N\big) - H\big(\mathbf{z}_1,\cdots,\mathbf{z}'_n,\cdots,\mathbf{z}_N\big)\Big| \leq c. \quad (14)$$

Then, we have for any $\xi > 0$

$$\Pr\Big\{H\big(\mathbf{z}_1,\cdots,\mathbf{z}_n,\cdots,\mathbf{z}_N\big) - \mathrm{E}H(\mathbf{z}_1,\cdots,\mathbf{z}_n,\cdots,\mathbf{z}_N) \geq \xi\Big\} \leq \exp\left\{N\Gamma\left(\frac{\xi}{Nc}\right)\right\}, \quad (15)$$

where $\Gamma(x)$ is defined in (13).

The inequality (15) is an extension of the classical Bennett's inequality and the two results will coincide if there is only one random variable as well.

Subsequently, we will use the above two deviation inequalities to obtain the generalization bounds based on the uniform entropy number and the Rademacher complexity, respectively.

4 Bennett-type Generalization Bounds

In this section, we will show the Bennett-type generalization bounds of the i.i.d learning process. The derived bounds are based on the uniform entropy number (UEN) and the Rademacher complexity, respectively. Note that since the presented bounds (16), (18) and (23) are derived from the Bennett-type deviation inequalities, they are said to be of Bennett-type as well.

4.1 UEN-based Generalization Bounds

By using the deviation inequality (12) and the symmetrization inequality [see Bousquet et al., 2004, Lemma 2], we can obtain a Bennett-type generalization bound based on the uniform entropy number:

Theorem 4.1 *Assume that $\mathcal{F}$ is a function class with the range $[a, b]$. Let $\mathbf{Z}_1^N$ and $\mathbf{Z'}_1^N$ be drawn from $\mathcal{Z}$ and denote $\mathbf{Z}_1^{2N} := \{\mathbf{Z}_1^N, \mathbf{Z'}_1^N\}$. Then, given any $0 < \xi \leq (b-a)$, we have for any $N \geq \frac{8(b-a)^2}{\xi^2}$,*

$$\Pr\left\{\sup_{f\in\mathcal{F}} \big|\mathrm{E}f - \mathrm{E}_N f\big| > \xi\right\} \leq 8\mathcal{N}_1\left(\mathcal{F}, \xi/8, 2N\right) \exp\left\{N\Gamma\left(\frac{\xi}{8(b-a)}\right)\right\}. \quad (16)$$

This theorem implies that the probability of the event that the generalization bound $\sup_{f\in\mathcal{F}} \big|\mathrm{E}f - \mathrm{E}_N f\big|$ is larger than any $\xi > 0$ can be bounded by the right-hand side of (16). We can find that the expression of the bound is similar to that of Bennett's inequality.

Different from the aforementioned Hoeffding-type result (7) and its alternative expression (8), it is difficult to directly achieve the alternative expression of the Bennett-type bound (16), because it is difficult to obtain the analytical expression of the inverse function of $\Gamma(x) = x-(x+1)\ln(x+1)$. Instead, one generally uses the term $\frac{-x^2}{2+(2x/3)}$ to approximate the function $\Gamma(x)$ and then get the so-called Bernstein's inequality.[5] In the same way, we can obtain the following alternative expression of the Bennett-type result (16):

$$\sup_{f\in\mathcal{F}} \big|\mathrm{E}_N f - \mathrm{E}f\big| \leq \frac{4(b-a)\big(\ln\mathcal{N}_1(\mathcal{F},\xi/8,2N) - \ln(\epsilon/8)\big)}{3N} + \frac{(b-a)\sqrt{2\big(\ln\mathcal{N}_1(\mathcal{F},\xi/8,2N) - \ln(\epsilon/8)\big)}}{\sqrt{N}}, \quad (17)$$

which implies that the rate of convergence of the Bennett-type bound is also up to $O\big(N^{-\frac{1}{2}}\big)$. It is in accordance with the rate of the aforementioned classical results (8). For convenience, the alternative expression is said to be of Bernstein-type if no confusion arises.

4.2 Bennett-type Alternative Expression and Faster Rate of Convergence

Recalling the process of obtaining Hoeffding's inequality and Bennett's inequality, the former is achieved by using the information of expectation, while the latter needs to consider the information of both expectation and variance. Intuitively, the Bennett-type result (16) should have a faster rate of convergence than that of the Hoeffding-type result (7). From this point of view, we introduce a new method to obtain another alternative expression of the Bennett-type result (16) and show that the rate of the generalization bound $\sup_{f\in\mathcal{F}} \big|\mathrm{E}_N f - \mathrm{E}f\big|$ can reach $o\big(N^{-\frac{1}{2}}\big)$, when N goes to *infinity*:

Theorem 4.2 *Follow the notations and conditions of Theorem 4.1. Then, given any $0 < \xi \leq (b-a)$ and for any $N \geq \frac{8(b-a)^2}{\xi^2}$, we have with probability at least $1-\epsilon$,*

$$\sup_{f\in\mathcal{F}} \big|\mathrm{E}_N f - \mathrm{E}f\big| \leq 8(b-a)\left(\frac{\ln\mathcal{N}_1\left(\mathcal{F},\xi/8,2N\right) - \ln(\epsilon/8)}{\beta_1 N}\right)^{\frac{1}{\gamma}}. \quad (18)$$

[5] http://ocw.mit.edu/courses/mathematics/18-465-topics-in-statistics-statistical-learning-theory-spring-2007/lecture-notes/l6.pdf

where $\beta_1 \in (0.0075, 0.4804)$,

$$\epsilon := 8\mathcal{N}_1\big(\mathcal{F}, \xi/8, 2N\big) \exp\left\{N\Gamma(x)\right\},$$

and $0 < \gamma\big(\beta_1; x\big) \leq \gamma < 2$ $(x \in (0, 1/8])$ *with*

$$\gamma(\beta; x) := \frac{\ln\Big(\big((x+1)\ln(x+1) - x\big)/\beta\Big)}{\ln x}. \qquad (19)$$

Proof. Since any $f \in \mathcal{F}$ is a bounded function with the range $[a, b]$, the supremum $\sup_{f\in\mathcal{F}} |\mathrm{E}f - \mathrm{E}_N f|$ will not be larger than $(b-a)$. Therefore, the quantity $\frac{\xi}{8(b-a)}$ in (16) will not be larger than $1/8$. Set $x = \frac{\xi}{8(b-a)}$ with $x \in (0, 1/8]$ and consider the following equation with respect to $\gamma > 0$

$$\Gamma(x) = x - (x+1)\ln(x+1) = -\beta_1(x^{\gamma}), \qquad (20)$$

where β_1 is some positive constant. Denote the solution to the equation (20) w.r.t. γ as

$$\gamma(\beta_1, x) := \frac{\ln\left(\frac{(x+1)\ln(x+1)-x}{\beta_1}\right)}{\ln(x)}. \qquad (21)$$

By numerical simulation, we find that given any $\beta_1 \in (0.0075, 0.4804)$, there holds that $0 < \gamma(\beta_1; x) < 2$ for any $x \in (0, 1/8]$ (see Fig. 1). Then, given any $x \in (0, 1/8]$ and $\beta_1 \in (0.0075, 0.4804)$, we have for any $\widetilde{\gamma} \in [\gamma(\beta_1; x), 2)$,

$$x - (x+1)\ln(x+1) \leq -\beta_1 x^{\widetilde{\gamma}} < -\beta_1 x^2. \qquad (22)$$

By combining Theorem 4.1 and (22), we can straightforwardly show an upper bound of the generalization bound $\sup\limits_{f\in\mathcal{F}} \big|\mathrm{E}_N f - \mathrm{E}f\big|$: letting

$$\epsilon := 8\mathcal{N}_1\left(\mathcal{F}, \xi/8, 2N\right) \exp\left\{N\Gamma(x)\right\},$$

and with probability at least $1 - \epsilon$,

$$\sup_{f\in\mathcal{F}} \big|\mathrm{E}_N f - \mathrm{E}f\big| \leq 8(b-a)\left(\frac{\ln\mathcal{N}_1(\mathcal{F}, \xi/8, 2N) - \ln(\epsilon/8)}{\beta_1 N}\right)^{\frac{1}{\gamma}}.$$

where $0 < \gamma\big(\beta_1; x\big) \leq \gamma < 2$ with $x \in (0, 1/8]$. ■

The above theorem shows a Bennett-type alternative expression of the generalization bound $\sup_{f\in\mathcal{F}} |\mathrm{E}_N f - \mathrm{E}f|$, which provides a faster rate $o(N^{-\frac{1}{2}})$ of convergence than the rate $O(N^{-\frac{1}{2}})$ of the classical result (8) and the Bernstein-type result (17). The main starting point of this theorem is whether there exists a positive constant β_1 such that the function $\gamma(\beta_1; x)$ is smaller than 2 for any $x \in (0, 1/8]$ [see (20) and (21)], *i.e.*, there holds that $0 < \gamma(\beta_1; x) < 2$ for any $0 < x \leq 1/8$. In the proof, we show that the value interval $(0.0075, 0.4804)$ of β_1 supports $\Gamma(x) = -\beta_1 x^{\gamma(\beta_1;x)} < -\beta_1 x^2$ for any $x \in (0, 1/8]$. As shown Fig. 1, given any $\beta_1 \in (0.0075, 0.4804)$, the function $\gamma(\beta_1; x)$ is smaller than 2 w.r.t. $x \in (0, 1/8)$. Meanwhile, given any $x \in (0, 1/8)$, the function $\gamma(\beta_1; x)$ is monotonically increasing w.r.t. $\beta_1 \in (0.0075, 0.4804)$. However, the function $\gamma(\beta_1; x)$ is not monotonically decreasing w.r.t. $x \in (0, 1/8)$ for any $\beta_1 \in (0.0075, 0.4804)$. In fact, $\gamma(\beta_1; x)$ is monotonically decreasing when $\beta_1 \in (0.0075, 0.4434]$ and has one single minimizer $x_0 \in (0, 1/8]$ of $\gamma(\beta_1; x)$ when $\beta_1 \in (0.4434, 0.4804)$ with $\gamma'(\beta_1; x_0) = 0$.

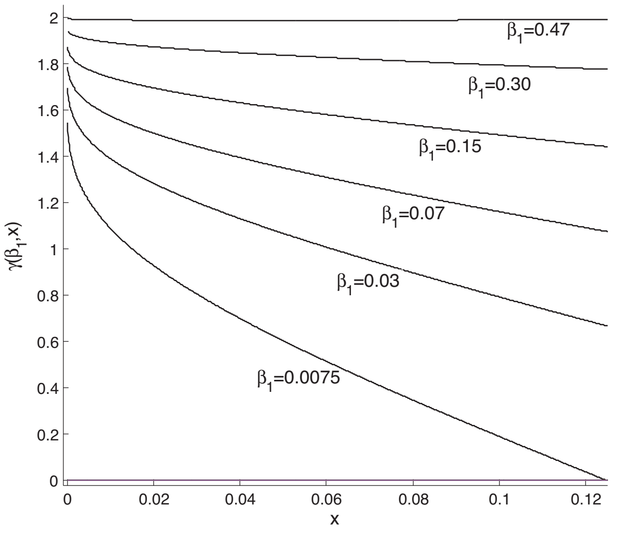


Figure 1: The Function Curve of $\gamma(\beta_1; x)$

4.3 Large-deviation Case

Subsequently, we will show that the Bennett-type bounds can reach a faster rate of convergence in the large-deviation case.

Remark 4.3 *The word "large-deviation" means that the discrepancy between the empirical risk and the expected risk is large (or not small). Given any* $\xi > 0$, *one of our major concerns is the probability* $\Pr\left\{\sup_{f\in\mathcal{F}} \big|\mathrm{E}_N f - \mathrm{E}f\big| > \xi\right\}$, *and then we say that the case the value of* ξ *approaches to* $(b-a)$ *is of large-deviation.*[6]

Actually, the large-deviation case is referring to a situation where the empirical quantity $\mathrm{E}_N f$ is far away from (at least not close to) the expected risk $\mathrm{E}f$. As shown in Fig. 1, for any $\beta_1 \in (0.0075, 0.4434]$, the curve of $\gamma(\beta_1; x)$ is monotonically decreasing w.r.t.

[6]The function class $\mathcal{F}$ is composed of bounded functions with the range $[a, b]$.

$x \in (0, 1/8]$, and for any $\beta_1 \in (0.4434, 0.4804)$ the function $\gamma(\beta_1; x)$ is still smaller than 2 for any $x \in (0, 1/8]$. Moreover, there holds that for any $\beta_1 \in (0.0075, 0.4804)$,

$$\lim_{x\rightarrow 0+} \gamma(\beta_1, x) := \lim_{x\rightarrow 0+} \frac{\ln\left(\frac{(x+1)\ln(x+1)-x}{\beta_1}\right)}{\ln(x)} = 2.$$

This illustrates that the rate $O(N^{-\frac{1}{\gamma}})$ $(\gamma(\beta_1; x) \leq \gamma)$ of the Bennett-type generalization bound (18) becomes faster as the empirical quantity $\mathrm{E}_N f$ goes further away from the expected risk $\mathrm{E}f$ (*i.e.* x approaches to 1/8). However, when $\mathrm{E}_N f$ approaches to $\mathrm{E}f$ (*i.e.* x goes to 0), the rate $O(N^{-\frac{1}{\gamma}})$ will approach to $O(N^{-\frac{1}{2}})$, which is in accordance with the classical Hoeffding-type results.

In contrast, the Hoeffding-type results consistently provide the rate $O(N^{-\frac{1}{2}})$ regardless of the discrepancy between $\mathrm{E}_N f$ and $\mathrm{E}f$. Thus, the Bennett-type results can give a more detailed description to the asymptotical behavior of the learning process.

4.4 Effect of the Parameter β_1

As addressed in the introduction, the main motivation of this paper is to study whether the Bennett-type generalization bounds have a faster rate of convergence than that of the Hoeffding-type generalization results, because the Bennett-type results are derived from the corporation of the expectation information and the variance information and in contrast, the Hoeffding-type results are only related to the expectation information. However, the main challenge to analyze the rate lies in obtaining the alternative expression of the bound (16), because it is difficult to analytically express the inverse function of $\Gamma(x)$. Instead, we exploit the term $-\beta x^\gamma$ to substitute $\Gamma(x)$ [see (20)] and Fig. 2 illustrates the validity of this method. In Fig. 2, the curve $\mathrm{e}^{-x^2/32}$ and $\mathrm{e}^{\Gamma(x/8)}$ correspond to the Hoeffding-type bound (7) and the Bennett-type bound (16), respectively. Evidently, setting $\beta_1 = 0.4804$ makes the curve $\mathrm{e}^{-\beta_1(x/8)^2}$ almost coincide with the curve $\mathrm{e}^{\Gamma(x/8)}$.

4.5 Rademacher-complexity-based Generalization Bounds

By using the deviation inequality (15), we can obtain the generalization bounds based on the Rademacher complexity:

Theorem 4.4 *Assume that $\mathcal{F}$ is a function class consisting of functions with the range $[a, b]$. Let $\mathbf{Z}_1^N = \{\mathbf{z}_n\}_{n=1}^N$ be a set of i.i.d. samples drawn from $\mathcal{Z}$. Then, for any $f \in \mathcal{F}$, we have with probability at least* $1 - \epsilon$,

$$\begin{aligned}\mathrm{E}f &< \mathrm{E}_N f + 2\mathcal{R}(\mathcal{F}) + (b-a)\left(\frac{\ln(1/\epsilon)}{\beta_2 N}\right)^{\frac{1}{\gamma}} \\ &< \mathrm{E}_N f + 2\mathcal{R}_N(\mathcal{F}) + 3(b-a)\left(\frac{\ln(2/\epsilon)}{\beta_2 N}\right)^{\frac{1}{\gamma}}, \quad (23)\end{aligned}$$

where β_2 is taken from the interval $(0.0075, 0.3863)$, $\epsilon := \exp\{N\Gamma(x)\}$ and $0 < \gamma(\beta_2; x) \leq \gamma < 2$ ($x \in (0, 1]$) with $\gamma(\beta; x)$ defined in (19)*.*

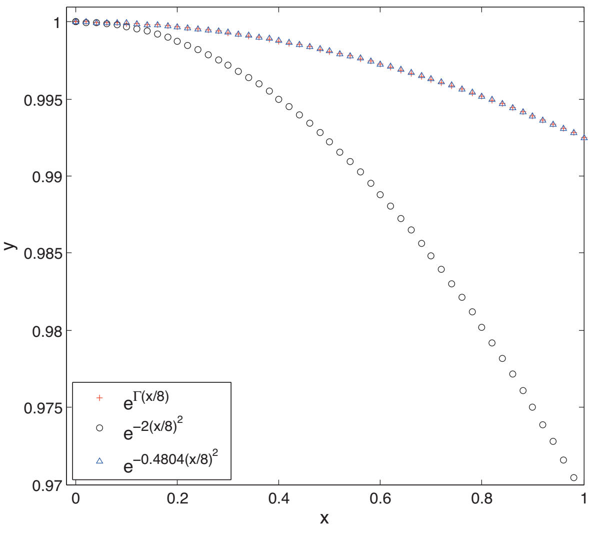


Figure 2: The Function Curves of $\mathrm{e}^{\Gamma(x/8)}$, $\mathrm{e}^{-x^2/32}$ and $\mathrm{e}^{-0.4804(x/8)^2}$.

The above theorem presents the Bennett-type generalization bounds based on the Rademacher complexity. Compared to the classical Rademacher-complexity-based results (9), the new bounds (23) incorporate a term $O\big(N^{-\frac{1}{\gamma}}\big)$ (*i.e.*, $o\big(N^{-\frac{1}{2}}\big)$ because $\gamma < 2$) with a faster rate of convergence than the term $O\big(N^{-\frac{1}{2}}\big)$ appearing in (9), when N goes to infinity.

5 Bounds of Rademacher Complexities

Recalling (16) and (23), it is observed that there might be a contradiction:

Remark 5.1 *The generalization bound* (16) *has the faster rate $o\big(N^{-\frac{1}{2}}\big)$, while the magnitude of the empirical Rademacher complexity has been proved to be $O\big(N^{-\frac{1}{2}}\big)$ as shown in the classical result* (10)*.*

In order to explain this contradiction, we should consider the following two questions:

(Q1) whether the empirical Rademacher complexity $\mathcal{R}_N(\mathcal{F})$ can totally describe the behavior of Rademacher complexity $\mathcal{R}(\mathcal{F})$;

(Q2) whether the classical results (10) are either applicable to $\mathcal{R}(\mathcal{F})$.

5.1 Answer to Question Q1

Recalling (9) and (23), the quantity $(\mathrm{E}f - \mathrm{E}_N f)$ is originally bounded by the Rademacher complexity $\mathcal{R}(\mathcal{F})$, while it is difficult to compute $\mathcal{R}(\mathcal{F})$ due to the unknown distribution of $\mathbf{Z}_1^N$. Instead, by using deviation inequalities (*e.g.* Hoeffding's inequality and Bennett's inequality), one can further bound $\mathcal{R}(\mathcal{F})$ by using its empirical version $\mathcal{R}_N(\mathcal{F})$ as follows: with probability as least $1 - \epsilon/2$,

$$\mathcal{R}(\mathcal{F}) \le \mathcal{R}_N(\mathcal{F}) + (b-a)\left(\frac{\ln(2/\epsilon)}{2N}\right)^{\frac{1}{2}},$$

and

$$\mathcal{R}(\mathcal{F}) \le \mathcal{R}_N(\mathcal{F}) + (b-a)\left(\frac{\ln(2/\epsilon)}{\beta_2 N}\right)^{\frac{1}{\gamma}}.$$

However, the behavior of Rademacher complexity $\mathcal{R}(\mathcal{F})$ cannot be totally described by its empirical version $\mathcal{R}_N(\mathcal{F})$. In fact, the joint distribution of $\sum \sigma_n f(\mathbf{z}_n)$ may not be a Rademacher process because $\mathbf{z}$ is a random variable of an unknown distribution.

5.2 Answer to Question Q2

Recalling (10), the lower and the upper bounds of $\mathcal{R}_N(\mathcal{F})$ are respectively derived from the Sudakov minoration for Rademacher processes and Massart's finite class lemma, both of which are strongly related to (or have the similar forms as that of) the Hoeffding-type results.[7] Thus, all Hoeffding-type conclusions mentioned in Section 2.3 are consistent.

However, as answered above, the joint distribution of $\sum \sigma_n f(\mathbf{z}_n)$ may not be a Rademacher process. Since Sudakov minoration is not valid for an arbitrary process [see Talagrand, 1994b,a, Latała, 1997], the lower bound of $\mathcal{R}_N(\mathcal{F})$ with the rate $O(N^{-\frac{1}{2}})$ is not applicable to the lower bound of $\mathcal{R}(\mathcal{F})$.

Additionally, we also need to consider the following question: why we do not use Bennett's inequality to obtain the upper bound of $\mathcal{R}_N(\mathcal{F})$? In fact, in order to obtain the upper bound of $\mathcal{R}_N(\mathcal{F})$, one needs to consider the following term $\mathrm{E}e^{\sigma f(\mathbf{z})}$. Since σ is a Rademacher variable taking values from $\{\pm 1\}$ with equivalent probability for a given sample $\mathbf{z}$, there holds that

$$\mathrm{E}e^{\sigma f(\mathbf{z})} = \frac{e^{f(\mathbf{z})} + e^{-f(\mathbf{z})}}{2} \le e^{\frac{(f(\mathbf{z}))^2}{2}},$$

which differs from the related formula (28) in Lemma A.1 that is built in the case that the distribution of $\mathbf{z}$ is unknown.

6 Asymptotical Convergence

Based on the generalization bound (16), we study the asymptotic convergence of the i.i.d. learning process. We also give a comparison with the existing results.

Recalling (13), it is noteworthy that there is only one solution $x = 0$ to the equation $\Gamma(x) = 0$ and $\Gamma(x)$ is monotonically decreasing when $x \ge 0$ [see Fig. 3]. Following Theorem 4.1, we can directly obtain the following result indicating that the asymptotic convergence of the i.i.d. learning process is determined by the uniform entropy number $\ln \mathcal{N}_1(\mathcal{F}, \xi/8, 2N)$.

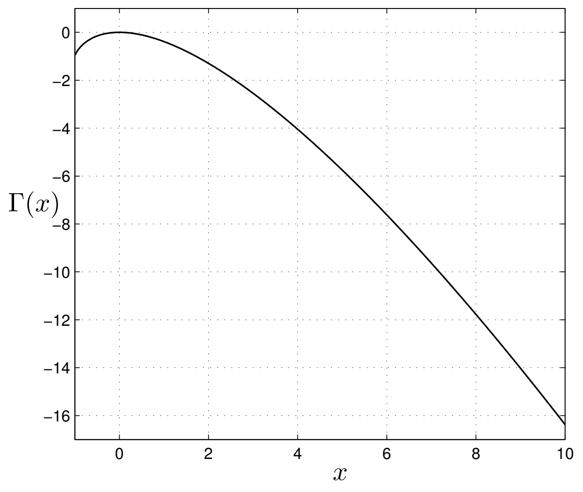


Figure 3: The Function Curve of $\Gamma(x)$

Theorem 6.1 *Under the notations and conditions of Theorem 4.1, if the following condition is supported:*

$$\lim_{N\to+\infty} \frac{\ln \mathcal{N}_1(\mathcal{F}, \xi/8, 2N)}{N} < +\infty, \tag{24}$$

then we have for any $\xi > 0$,

$$\lim_{N\to+\infty} \Pr\left\{\sup_{f\in\mathcal{F}} \left|\mathrm{E}f - \mathrm{E}_N f\right| > \xi\right\} = 0. \tag{25}$$

[7]Recalling Massart's finite class lemma, the main step of its proof is processed by using a term $e^{\lambda^2 r^2/2}$ which is also similar to the term $e^{\lambda^2 r^2/8}$ appearing in Hoeffding's lemma.
See http://ttic.uchicago.edu/∼tewari/lectures/lecture10.pdf

As shown in Theorem 6.1, if the uniform entropy number $\ln \mathcal{N}_1(\mathcal{F}, \xi/8, 2N)$ satisfies the condition (24), the probability of the event

$$\sup_{f\in\mathcal{F}} \big|\mathrm{E}f - \mathrm{E}_N f\big| > \xi$$

will converge to *zero* for any $\xi > 0$, when the sample number N goes to *infinity*. This is in accordance with the classical result shown in (7): the probability of the event that $\sup_{f\in\mathcal{F}} \big|\mathrm{E}f - \mathrm{E}_N f\big| > \xi$ will converge to *zero* for any $\xi > 0$, if the uniform entropy number $\ln \mathcal{N}_1(\mathcal{F}, \xi/8, 2N)$ satisfies the same condition as (24).

7 Conclusion

In this paper, we present the Bennett-type generalization bounds and analyze the rate of convergence of the derived bounds. In particular, we first extend the classical Bennett's inequality to develop two Bennett-type deviation inequalities. Based on the derived inequality, we then obtain the generalization bounds based on the uniform entropy number and the Rademcaher complexity, respectively.

Moreover, we show that the Bennett-type generalization bounds have a faster rate $o(N^{-\frac{1}{2}})$ of convergence than the rate $O(N^{-\frac{1}{2}})$ of the classical Hoeffding-type results [see (7) and (8)]. Especially, we show that the rate will become faster in the large-deviation case, where the empirical risk $\mathrm{E}_N f$ is far away from the expected risk $\mathrm{E}f$. In contrast, the Hoeffding-type results provide the rate $O(N^{-\frac{1}{2}})$ regardless of the discrepancy between $\mathrm{E}_N f$ and $\mathrm{E}f$. From view of this point, the Bennett-type results give a more detailed description to the asymmetrical behavior of the learning process.

We give an explanation to the "contradiction" mentioned in Remark 5.1. As shown in Talagrand [1994b,a], Latała [1997], Sudakov minoration provides a lower bound of the empirical Rademacher complexity $\mathcal{R}_N(\mathcal{F})$ with the rate $O(N^{-\frac{1}{2}})$ [Mendelson, 2003], because the empirical Rademacher complexity $\mathcal{R}_N(\mathcal{F})$ actually is a special case of subgaussian process given a sample set $\{\mathbf{z}_n\}_{n=1}^N$. In contrast, it is difficult to specify the distribution characteristics of $\mathcal{R}(\mathcal{F})$ and thus the classical result (10) could not be applied to bound $\mathcal{R}(\mathcal{F})$.

References

P.L. Bartlett, O. Bousquet, and S. Mendelson. Local rademacher complexities. *Annals of Statistics*, 33 (4):1497–1537, 2005.

S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J.W. Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1): 151–175, 2010.

G. Bennett. Probability inequalities for the sum of independent random variables. *Journal of the American Statistical Association*, 57(297):33–45, 1962.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

O. Bousquet. A bennett concentration inequality and its application to suprema of empirical processes. *Comptes Rendus Mathematique*, 334(6):495–500, 2002.

O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. *Advanced Lectures on Machine Learning*, pages 169–207, 2004.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

R. Latała. Sudakov minoration principle and supremum of some processes. *Geometric & Functional Analysis GAFA*, 7(5):936–953, 1997.

S. Mendelson. A few notes on statistical learning theory. *Advanced Lectures on Machine Learning*, pages 1–40, 2003.

M. Mohri and A. Rostamizadeh. Stability bounds for stationary ϕ-mixing and β-mixing processes. *Journal of Machine Learning Research*, 11:798–814, 2010.

Michel Talagrand. Constructions of majorizing measures bernoulli processes and cotype. *Geometric & Functional Analysis GAFA*, 4(6):660–717, 1994a.

Michel Talagrand. The supremum of some canonical processes. *American Journal of Mathematics*, 116 (2):283–325, 1994b.

A. Van der Vaart and J. Wellner. *Weak Convergence and Empirical Processes: with Aapplications to Statistics*. Springer, 1996.

V.N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

AUAI Press
P.O. Box 866
Corvallis, Oregon 97339
USA