Zeroth Order Non-convex optimization with Dueling-Choice Bandits

Yichong Xu, Aparna Joshi, Aarti Singh, Artur Dubrawski
Machine Learning Department, Carnegie Mellon University

Abstract
We consider a novel setting of zeroth order non-convex optimization, where in addition to querying the function value at a given point, we can also duel two points and get the point with the larger function value. We refer to this setting as optimization with dueling-choice bandits, since both direct queries and duels are available for optimization. We give the COMP-GP-UCB algorithm based on GP-UCB (Srinivas et al., 2009), where instead of directly querying the point with the maximum Upper Confidence Bound (UCB), we perform constrained optimization and use comparisons to filter out sub-optimal points. COMP-GP-UCB comes with theoretical guarantee of $O(\frac{\Phi}{\sqrt{T}})$ on simple regret where $T$ is the number of direct queries and $\Phi$ is an improved information gain stemming from a comparison-based constraint set that restricts the space for optimum search. In contrast, in the plain direct query setting, $\Phi$ depends on the entire domain. We discuss theoretical aspects and show experimental results to demonstrate efficacy of our algorithm.

1 Introduction
Zeroth order non-convex optimization, also variously known as continuous multi-armed bandit or black-box optimization, is an important problem that naturally appears in various domains like dynamic pricing (Besbes and Zeevi, 2009), reinforcement learning (Smart and Kaeltbling, 2000) and material science (Xue et al., 2016). With an unknown black-box function $f: \mathcal{X} \rightarrow \mathbb{R}$, zeroth order optimization aims to find the optimal point of the function with as few queries to (a noisy version of) $f(x)$ as possible, with no gradient information directly available. Although zeroth order convex optimization is generally efficient (Jamieson et al., 2012), optimizing a non-convex $f$ under smoothness constraints requires the same effort as estimating $f$ almost everywhere, and usually leads to a query complexity exponential in $d$, where $d$ is the feature space dimensionality (Chen, 1988; Flaxman et al., 2005; Ge et al., 2015; Wang et al., 2018). The prohibitive cost for non-convex optimization has motivated research on suitable assumptions, such as linear bandits (Rusmevichientong and Tsitsiklis, 2010), convex approximations (Wang et al., 2018), and optimization based on level sets (Malherbe et al., 2016; Wang et al., 2018). We propose a complementary approach, where in addition to direct queries to $f$, one can also compare two points in feature space, and obtain the point with a larger $f$ value. Inspired by dueling bandits in the bandit domain (Yue et al., 2012), we call our setting non-convex optimization with dueling-choice bandits; we note that different than dueling bandits, here direct queries and comparisons are both available for optimizing $f$, and therefore duels are available as an additional choice.

In many applications, comparisons can be available at a cheaper price than direct queries. For example in preference elicitation, the user can give scores on recommended items, as well as (more easily) compare two items to choose the preferred one. Similarly, for hyperparameter search of information retrieval (IR) models, direct queries typically involve collecting relevance scores from paid workers, whereas comparisons can be obtained by interleaving the ranking of two different models, and observing user click on the retrieval results (Radlinski et al., 2008). Such comparisons usually come with less cost in both time and money. As another example, in material synthesis, we aim to optimize the desired properties of materials by controlling the input parameters (temperature, pressure, etc.) (Faber et al., 2016; Xue et al., 2016). While material synthesis is expensive, comparisons can be carried out by asking material scientists.

Related Work. There is a vast literature on zeroth order non-convex optimization (Bull, 2011; Chen, 1988; Flaxman et al., 2005; Ge et al., 2015; Hazan et al., 2017; Wang et al., 2018). We build our work on GP-UCB (Srinivas et al., 2009), a method for optimizing unknown functions under the Gaussian process (GP) assumption by optimiz-
ing the Upper Confidence Bound (UCB). Closest to our setting is a line of recent research on multi-fidelity GP optimization (Kandasamy et al., 2016, 2017; Sen et al., 2018), which assumes that we can query the target functions at multiple fidelities of different costs and precision. We detail the relation and difference of our setting with multi-fidelity optimization in Section 3.6. To briefly describe it, our setting is harder since we cannot directly query the function on which comparisons are based. Moreover, the multi-fidelity assumptions such as fidelities being close in sup-norm do not hold for our setting since any constant shift of the comparison function yields the same comparisons. We instead consider an active transfer learning setting where information from a function that can be learned using comparisons is transferred actively to optimize the target function (refer to Section 2 for details).

Optimization with comparisons has been studied under the framework of derivative-free optimization (Jain et al., 2012; Kumagai, 2017) and continuous dueling bandits (Ailon et al., 2014; Sui et al., 2017). Kumagai (Kumagai, 2017) obtains optimal regret rates for a convex $f$. However to the best of our knowledge, no previous work has theoretical guarantees on optimizing a non-convex $f$. Also, these results cannot be applied when the comparisons are biased (i.e., a Condorcet winner on comparisons might not be the best point for direct queries).

Finally, there is another line of research that combines direct queries and comparisons for classification or regression problems (Kane et al., 2017; Xu et al., 2017, 2018). Our methods differ from theirs because we focus on the optimization setting, and only care about points near the optimum. These methods make direct queries across the whole feature space to learn the underlying function well, which is unnecessary for optimization.

Our Contributions. We develop and evaluate a new algorithm for non-convex optimization with dueling choices, which we refer to as Comparison-based GP-UCB (COMP-GP-UCB). Our theoretical and experimental results show the strengths of our algorithm.

- When we can obtain comparisons based on the target function $f$, we show that comparisons can be as powerful as direct queries: COMP-GP-UCB can achieve the same rate of convergence as its label-only counterparts, while using only comparisons and no direct queries. This solves the open problem raised in Sui et al. (2017), to develop continuous dueling bandit algorithms with no-regret guarantees.

- Next, we assume that comparisons are based on a misspecified function $f_c$, where $f_c$ approximates $f$. COMP-GP-UCB in this case uses comparisons to optimize a function $f_c$ which has the same optimizer as $f_c$, and then use direct queries to search in a smaller region for the optimum of the target function. The regret rate of COMP-GP-UCB is then better than the label-only counterparts, and it depends on the difference between $f_c$ and $f$: the better the approximation, the lower the regret we can get from COMP-GP-UCB. We further demonstrate a version of the algorithm that adapts to this difference. Our algorithm also extends multi-fidelity GP optimization to the setting where information is transferred actively from a lower fidelity to a higher fidelity while only assuming that the optimizer of the lower fidelity (source function) is within a constant distance of the optimizer of the higher fidelity (target function), instead of the fidelities being close everywhere.

- In our experiments, we test COMP-GP-UCB on multi-fidelity functions from previous literature and show that it outperforms label-only algorithms and existing multi-fidelity algorithms when comparisons are cheaper than direct labels.

2 Background and Problem Setup

We aim to maximize a function $f : \mathcal{X} \rightarrow \mathbb{R}$, where $\mathcal{X} \subseteq [0, r]^d$ is the feature space. In each iteration $t$ of optimization, we can query (expensive) direct queries to $f$ at a chosen point $x_t$, and obtain $y = f(x_t) + \varepsilon$, $\varepsilon \in [-\eta, \eta]$ and $E[\varepsilon] = 0$, with $\eta > 0$ a known constant.

Comparison Probabilities. In addition to traditional direct queries $y$, we can obtain (cheap) comparisons for a pair of points $(x, x') \in \mathcal{X} 	imes \mathcal{X}$. We assume that comparisons are based on a function $f_c$ which can be potentially different from $f$ (as described later in this section). A common assumption in the literature is to use a link function to assume a distribution of the comparisons, i.e., we assume $\text{Pr}(x \succ x') = \sigma(f_c(x) - f_c(x'))$ for some function $\sigma$. Common link functions include logistic function (BTL model (Bradley and Terry, 1952)), or Gaussian cdf (Thurstone model (Thurstone, 1927)).

Connecting comparisons and direct queries. To make comparisons helpful for optimization, we also require that $f_c$ is a good approximation of $f$. Here we differentiate between two settings:

- Dueling-Choice Bandits with unbiased comparisons: We assume comparison comes from the same function as the target function, i.e., $f_c = f$ or, more generally, that $f_c$ and $f$ have the same optimizer ($\zeta = 0$ as described below). This may be the case when comparison and direct queries come from the same agent, such as the preference elicitation example in the introduction.

Our methods can also be extended to the setting where $\varepsilon$ follows a sub-Gaussian distribution with parameter $\eta$. We assume a bounded $\varepsilon$ for simplicity here.
• Dueling-Choice Bandits with misspecified comparisons: We assume $f_r \approx f$. In many cases, comparisons are from a different source (e.g. experts) than direct queries (e.g. experiments), and this can result in a biased $f_r$. To this end, we assume a bounded difference near the optimum:

**Assumption 1.** Let $f^* = \max_x f(x)$ and $f^*_r = \max_x f_r(x)$. There exists a constants $\xi$ such that for any point $x \in \mathcal{X}$ we have $|(f^*_r - f_r(x)) - (f^* - f(x))| \leq \xi$.

In words, when we get $\varepsilon$-close to the maximum of $f$, we are at least $(\varepsilon + \xi)$-close to the maximum of $f_r$, and vice versa. Under this assumption, we would require both comparison and direct queries if we want to achieve optimization error smaller than $\xi$.

We note that our results can be generalized to the case where Assumption 1 only holds for $f_r \approx f$.

**Budgets and Regrets.** We analyze the problem of optimizing $f$ under a cost budget $\Lambda$. Suppose a direct query costs $\lambda_1$ units of some resource and a comparison costs $\lambda_c < \lambda_1$. Also, let $\nu_\Lambda = \{ A \}$ be the upper bound on number of queries when we use all the budget on comparisons, and $\nu_\Lambda = \{ A \}$ be the corresponding lower bound when we only use direct queries. Also let $q_{\text{label}}$ if we make direct queries at iteration $t$, and $q_{\text{comp}}$ otherwise. We analyze the simple regret under budget $\Lambda$, defined as follows:

\[
S(\Lambda) = \min_t r_t
\]

\[
= \min_t \left\{ f^* - f(x_t), \min \{ f^* - f(x_t), f^* - f(x'_t) \} \right\}, \quad \text{if } q_t = \text{label},
\]

\[
\min \{ f^* - f(x_t), \min \{ f^* - f(x_t), f^* - f(x'_t) \} \}, \quad \text{if } q_t = \text{comp}.
\]

In words, we calculate the minimum regret achieved by either comparison or direct queries. We compute simple regret over all direct queries; for comparisons, we adopt the notion of weak regret employed in (Yue et al., 2012).

Here we choose simple regret because our target is to optimize function $f$, and cumulative regret is typically not relevant for our setting. Our method can also be easily extended to the optimizer error setting, where the algorithm gives an estimation of the optimum when it ends. In analyzing the regret rates, we use $O(\cdot)$ to ignore constants, and $O(\cdot)$ to ignore log terms in the regret bounds.

### 3 Algorithm and Analysis

We describe our COMP-GP-UCB algorithm in this section. We first present the Gaussian process framework on which our algorithm is based in Section 3.1. Then we present the algorithm in Section 3.3 under the assumption that $\zeta$ is known. This includes the unbiased comparison case, where $\zeta = 0$. We present our theoretical analysis for COMP-GP-UCB in Section 3.4. Finally, we give an extension to adapt to unknown $\zeta > 0$ in Section 3.5.

#### 3.1 The Gaussian Process Back End

We base our methods on Gaussian Process, with kernel function $\kappa$. If $f$ was sampled from the Gaussian process $\mathcal{GP}(0, \kappa)$, and the direct queries were coming from $f$ plus a Gaussian noise, i.e., $D = \{(x_i, y_i)\}_{i=1}^n$ with $y_i = f(x_i) + \varepsilon_i$, $\varepsilon_i \sim \mathcal{N}(0, \eta^2)$, then the posterior distribution at $f(x)|D$ would be a Gaussian $\mathcal{N}(\mu_t(x), \sigma_t(x))$ with

\[
\mu_t(x) = k^T(K + \eta^2 I_n)^{-1}Y,
\]

\[
\sigma_t(x) = \kappa(x, x) - k^T(K + \eta^2 I_n)^{-1}k.
\]

Here $Y = (y_1, \ldots, y_n)^T$, $k = (\kappa(x, x_1), \ldots, \kappa(x, x_t))^T$, and matrix $K \in \mathbb{R}^{t \times t}$ is given by $K_{ij} = \kappa(x_i, x_j)$, and $I_n$ is the $n \times n$ identity matrix.

**Remark.** We note that the Gaussian noise and prior is only assumed to derive updates to the mean and variance in the algorithm, and we do not assume the actual feedback follow a Gaussian model, nor the functions are sampled from the Gaussian process. We only assume that $f$ have bounded norm in $\mathcal{H}_\kappa$, and that $\varepsilon$ is bounded in $[-\eta, \eta]$, as stated in Section 2. This is the same as the agnostic setting in GP-UCB (Chowdhury and Gopalan, 2017; Kandasamy et al., 2016).

**The Maximum Information Gain.** As in previous works on GP (Chowdhury and Gopalan, 2017; Kandasamy et al., 2016), our results will depend on the maximum information gain (Srinivas et al., 2009) between function measurements and the function values, defined as below:

**Definition 1.** Suppose $A \subseteq \mathcal{X}$ is a subset of feature space, and $A = \{ x_1, \ldots, x_n \} \subseteq A$ is a finite subset of $A$. Then the maximum information gain on $A$ with $n$ evaluations is defined as $\Phi_n(A) = \max_{A \subseteq A | A = n} \{ f(A) + \varepsilon \tilde{A}; f(A) \}$, where $f(A) = \{ f(x) \}_{x \in A}$ $\varepsilon \tilde{A} \sim \mathcal{N}(0, \eta^2 I)$, and $I(X, Y) = H(X) - H(Y | X)$ is the mutual information.

When $\mathcal{X} \subseteq \mathbb{R}^d$ is compact and convex, Srinivas et al. (2009) shows that i) for linear kernel $\kappa$, $\Phi_n(\mathcal{X}) = O(d \log n)$; ii) for squared exponential (SE, or RBF) kernel, $\Phi_n(\mathcal{X}) = O((\log n)^{d+1})$; iii) For Matérn kernels $\kappa(x, x') = \nu^{-\nu} (\frac{2\nu}{\nu-\nu})^{\nu} B_{\nu}(\frac{2\nu}{\nu-\nu})$, we have $\Phi_n(\mathcal{X}) = O\left( \frac{d(d+1)}{n^{\nu-\nu+1}} \log n \right)$.

**Review of GP-UCB and IGP-UCB.** Previous sequential optimization has adopted the upper confidence bound
We instead consider a different function directly related to
An advantage of using \( f \) is a straightforward way to incorporate comparisons into
the UCB principle, where we maintain a high-confidence
maximum value of \( f \) from \( t - 1 \), and \( \beta \) is a multiplier that increases with \( t \). We describe
these algorithms in detail in Appendix.

3.2 The Borda Function \( f_c \)
A straightforward way to incorporate comparisons into
optimization is to use them to compute a GP posterior of
either \( f \) or \( f_c \). However, we will face several difficulties.
Firstly, the posterior based on comparisons cannot be
analytically computed. Also, we cannot compute the joint
posterior based on both direct queries and comparisons,
since \( f \) and \( f_c \) can be different. Lastly, comparisons might
not be truthful and can be inconsistent; i.e., human might
give contradicting comparisons like \( x_1 \succ x_2 \succ x_3 \succ x_1 \).

We instead consider a different function directly related to
\( f_c \), defined as \( f_c(x) = \Pr[x \succ X] \), where \( X \) is randomly
chosen from \( X \). In words, \( f_c(x) \) is the probability that
\( x \) beats a random point \( X \in X \). We refer to \( f_c \) as the
Borda function, inspired by Borda scores in the dueling
bandits literature (Heckel et al., 2016; Zoghi et al., 2015).
An advantage of using \( f_c \) is that we can obtain unbiased
estimates of \( f_c(x) \) by comparing \( x \) to a random point in
\( X \in X \).

It is easy to see that \( f_c \) should have the same optimizer
as \( f \). We make the following assumption to ensure
usefulness of comparisons:

**Assumption 2.** Let \( f^*_c = \max_x f_c(x) \) and \( f^*_c = \max_x f_c(x) \). There exists constants \( L_1, L_2 \) such that for every \( x \in X \) we have \( \frac{1}{\sqrt{t}} (f^*_c - f_c(x)) \leq f^*_c - f_c(x) \leq L_2 (f^*_c - f_c(x)) \).

In other words, difference in \( f_c \) will cause a difference of
similar scale in \( f_c \). This requires that the comparisons
induces a Borda function \( f_c \) such that \( f_c \) is close to \( f_c \) at
its optimum, and that \( f_c \) and \( f_c \) have the same optimizer.
We note that this is a quite weak assumption, as we do not
restrict the result of comparing individual points \( x, x' \)
to comply with \( f_c(x) \approx f_c(x') \), i.e., comparisons do not
need to be consistent. We can show that Assumption 2 holds
under the link function setting, when \( \sigma \) is Lipschitz
continuous:

**Proposition 1.** Suppose comparisons follows a link function \( \sigma \) with a Lipschitz constant between \([1/L_1, L_2] \), i.e.,
\[ \frac{\sigma(x) - \sigma(y)}{|x - y|} \in [\frac{1}{L_1}, L_2], \forall x, y \in \mathbb{R}, \] then Assumption 2 holds.

We comment that common link functions such as BTL
(Bradley and Terry [1952] and Thurstone [Thurstone
1927]) all have bounded Lipschitz functions if \( f_c \) is bounded.
Lastly, we note that Ailon et al. (2014) also compare
\( x \) to a random point \( X \), and use the feedback to update
the function estimates. However, their method relies on a
linear link function \( \sigma(x) = \frac{1}{2}x \) and cannot be applied
for BTL or Thurstone models.

**Algorithm 1 COMP-GP-UCB**

**Input:** Comparison bias \( \zeta \), comparison exploration
threshold \( \gamma \), confidence \( \delta \)

1: Set \( D_0^t = D_0^t = 0 \), \( (\mu_0^{(r)}, \sigma_0^{(r)}) = (\mu_0^{(l)}, \sigma_0^{(l)}) = (0, \kappa^{1/2}) \), \( t \leftarrow 0 \)

2: **repeat**

3: **Compute** \( x_1 = \arg \max_{x \in X} \mu_t^{(r)}(x) + \beta_t^r \sigma_t^{(r)}(x) \)

4: **QUERY** \( x_i \), \( \text{comp} \)

5: **until** \( \beta_t^r \sigma_t^{(r)}(x_t) \leq \gamma \) or budget exhausted

6: Let \( \tilde{f}_r = \mu_t^{(r)}(x_i) - \beta_t^r \sigma_t^{(r)}(x_t) \)

7: while Budget not exhausted do

8: Let \( \phi_t^{(r)}(x) = \mu_t^{(r)}(x) + \beta_t^r \sigma_t^{(r)}(x) - f_c + \frac{1}{\sqrt{t}} \)

9: Compute \( x_t = \arg \max_{x \in X, x \neq X} \phi_t^{(r)}(x) \geq 0 \mu_t^{(l)}(x) + \beta_t \sigma_t^{(l)}(x) \)

10: if \( \beta_t^r(x_t) \sigma_t^{(r)}(x_t) \geq \gamma \) then

11: else Query \((x_t, \text{label})\)

12: end if

13: end while

14: **end while**

15: **procedure** Query(query point \( x_t \), query type \( q_t \))

16: if \( q_t = \text{comp} \) then

17: Sample \( x' \) randomly from \( X \) and query to

18: Update \( D_t^c \leftarrow D_{t-1}^c \cup \{ (x_t, z_t) \} \), \( D_t^l \leftarrow \)

19: Perform Bayesian update for \( \mu_t^{(r)}, \sigma_t^{(r)} \) based

20: else

21: Query direct labels for \( x_t \) and obtain \( y_t \)

22: Update \( D_t^l \leftarrow D_{t-1}^l \cup \{ (x_t, y_t) \} \), \( D_t^c \leftarrow \)

23: Perform Bayesian update for \( \mu_t^{(l)}, \sigma_t^{(l)} \) based

24: end if

25: **end procedure**
3.3 Optimization with Known \( \zeta \)

When \( \zeta \) is known and given, COMP-GP-UCB is formally described in Algorithm 1. Our algorithm works both for unbiased comparisons (\( \zeta = 0 \)) and misspecified comparisons (\( \zeta > 0 \)). COMP-GP-UCB is an anytime algorithm, meaning that it does not need to know the total budget \( \Lambda \) before it begins. For any input \( \zeta \geq 0 \), the high-level idea is to constrain the search region for \( f \) using comparisons to the set \( \mathcal{H} := \{ x : f_r(x) \geq f^* - L_2 \zeta \} \) where \( f^* = \max_x f_r(x) \). \( \mathcal{H} \) is guaranteed to contain the optimizer \( f \) under our assumptions; to see this, let \( x^* \) be any optimizer of \( f \), and we have \( f^* - f_r(x^*) \leq L_2(f^* - f(x^*)) \leq L_2(f^* - f(x^*)) + \zeta = L_2 \zeta \). The first inequality follows from Assumption 2 and the second one follows from Assumption 1. It is easy to see that \( \mathcal{H} \) is much smaller than \( \tilde{X} \) if comparisons are mostly correct (i.e., \( \zeta \) is small); therefore we can explore more efficiently by restricting the search on \( \mathcal{H} \).

COMP-GP-UCB takes as input \( \zeta \), a parameter \( \gamma \) to control exploration on comparisons, and a confidence level \( \delta \). We keep track of posteriors \( (\mu^{(l)}) \), \( (\sigma^{(l)}) \) for \( f \) and \( f_r \), and construct confidence intervals \( \mu_{t-1}^{(l)}(x) \pm \beta \sigma_{t-1}^{(l)}(x), \mu_{t-1}^{(r)}(x) \pm \beta \sigma_{t-1}^{(r)}(x) \). Since \( f_r \) is unknown, to approximate \( \mathcal{H} \), the algorithm adopts a two-phase approach: In the first phase (Step 1-5), we optimize \( f_r \) using comparison queries until \( \beta \sigma_{t-1}^{(r)}(x_t) \leq \gamma \), i.e., the queried point has confidence of at least \( \gamma \). At the end of the first phase, we compute \( f_r \) as a lower bound for \( f^* \). Next, we start the second phase exploring \( f \) (Step 6-13). We select the query point \( x_t \) based on a filtering \( \phi^{(r)}(x) \geq 0 \); the filtering approximates the constraint set \( \mathcal{H} \) by combining the current UCB of \( f_r \) and the LCB \( f_r \) from the first phase. Then the algorithm optimizes the UCB of \( f \) under the constraint of \( \phi^{(r)}(x) \geq 0 \). While doing this, we check the UCB of \( f_r \) at the maximizer \( x_t \) and if we are not confident about \( f_r(x_t) \), we query a comparison, or otherwise we make a direct query as in GP-UCB.

The query process is described in the procedure QUERY. For direct queries, we query \( x_t \) directly, and update the posterior of \( f \) according to (2); for comparisons, we compare \( x_t \) with a random point \( x' \), and use the result as feedback to update posterior of \( f_r \). We note that this comparison result is an unbiased estimate of \( f_r(x_t) \).

The Two-Phase Approach. Both phases are critical for the algorithm to succeed. The first phase is important in two ways: Firstly, it helps to get a low regret in the unbiased comparisons setting, and in the initial stages of the algorithm when only comparison queries are used for the biased (misspecified comparison) setting. Also, it gives a lower bound \( f_r \leq f^* \) of the optimum of \( f_r \) at Step 6 which will be used to approximate the constraint set \( \mathcal{H} \). Then we use the second phase to obtain low regret in the biased comparison case.

Choice of \( \phi^{(r)}(x) \). The choice of \( \phi^{(r)}(x) \) is critical for the algorithm to succeed. We want that the region \( S = \{ x : \phi^{(r)}(x) \geq 0 \} \) is not too small or too large: we need that every maximizer \( x^* \) of \( f \) is in \( S \), while also excluding as many points as possible using the information from \( f_r \). To achieve the former, we have added \( L_2 \zeta \) to the confidence interval to account for the difference in \( f_r \) and \( f \). To achieve the latter, we need both a good UCB of \( f_r \) and a good LCB of \( f^* = \max_x f_r(x) \). The good UCB is ensured by the check at Step 10; we only make direct queries when we are confident enough about \( f_r(x_t) \). The good LCB is ensured by the first phase, where we compute \( f_r \); without the first phase \( f_r \) can be arbitrarily bad and it will lead to suboptimal direct queries. In the proof we show that when \( \phi^{(r)}(x) \geq 0 \) and \( \beta \sigma_{t-1}^{(r)}(x) \geq \gamma, x \) belongs to an approximation of \( \mathcal{H} \). So the two constraints combined ensure that we use direct queries to explore \( \mathcal{H} \).

3.4 Theoretical Analysis

We now present our theoretical results. We defer full proofs to the appendix due to space constraints. We first analyze the unbiased comparison case. In this case, we have \( \zeta = 0 \), and we only need comparisons to achieve low regret. Therefore we run COMP-GP-UCB with \( \zeta = \gamma = 0 \); in this case, the algorithm only executes the first phase, and only uses comparisons to optimize \( f_r \). We obtain the following guarantee.

**Theorem 2.** Suppose Assumption 2 holds, and \( f_r = f \). Let \( \beta^{(r)} = 2B + \sqrt{2(\Phi_{t-1}(X) + 1 + \log(1/\delta))} \). There exists a constant \( C \) dependent on \( d, \kappa \) such that COMP-GP-UCB with \( \zeta = \gamma = 0 \) has a simple regret bounded by

\[
S(\Lambda) \leq C \left( B + \sqrt{\left( \Phi_{t-1}(X) + \log(1/\delta) \right)} \right) \sqrt{\frac{\Phi_{\pi_{t-1}}(X)}{\pi_{t-1}}}.
\]

**Remark.** IGP-UCB (Chowdhury and Gopalan 2017) in the label-only setting has regret of form

\[
S_{IGP-UCB}(\Lambda) \leq C \left( B + \sqrt{\left( \Phi_{t-1}(X) + \log(1/\delta) \right)} \right) \sqrt{\frac{\Phi_{\pi_{t-1}}(X)}{\pi_{t-1}}},
\]

where \( \pi_{t-1} = \left\lfloor \frac{\Lambda}{\lambda} \right\rfloor \). This is the same form as (3), but with \( \pi_{t-1} \) replaced with \( \pi_{t-1} \). Recall that \( \pi_{t-1} \) is the number of queries when we use all the budget on comparisons, and \( \pi_{t-1} \) is the number for using all budget on direct queries. In other words, COMP-GP-UCB has the same rate as IGP-UCB as if direct queries are as cheap as comparisons. When comparisons are much cheaper than direct queries, COMP-GP-UCB leads to a great advantage by
significantly reducing the number of direct queries needed.

We then analyze COMP-GP-UCB in the misspecified comparison setting ($\zeta > 0$). In this setting, comparisons act as a filter on $\mathcal{X}$ to reduce the search region for direct queries. When $f_c$ approximates $f$ well (i.e., a small $\zeta$), the set $\mathcal{H} = \{x : f_c(x) \geq f^*_r - L_2\zeta\}$ is much smaller than the feature space $\mathcal{X}$. Therefore by using comparison terms, we wish to replace the $\Phi_{\eta}(\mathcal{X})$ term in (4) by $\Phi_{\eta}(\mathcal{H})$, effectively exploring a smaller region. We show that COMP-GP-UCB can have a similar behavior by exploring on a slightly larger set dependent on $\gamma$, defined as $\mathcal{H} = \{x \in \mathcal{X} : f_c(x) \geq f^*_r - L_2\zeta - 4\gamma\}$. The following theorem characterizes the regret of COMP-GP-UCB under the misspecified comparison setting.

**Theorem 3.** Suppose Assumptions 2 and 4 hold, and $\zeta$ is known. Let $\beta_i^{(r)}$ be the same as in Theorem 2 and $\beta_i = 2B + \sqrt{2(\Phi_{\eta}(\mathcal{H}) + 1 + \log(1/\delta))}$. There exists constants $\Lambda_0$, $C$ dependent on $\zeta$, $\gamma$, $B$, $d$, $\kappa$ such that if when $\Lambda \geq \Lambda_0$ we have $S(\Lambda) \leq \min\{S_1, S_2\}$, where

\[
S_1 = 2L_1\gamma + \zeta + C\left(B + \sqrt{(\Phi_{\eta}(\mathcal{X}) + \log(1/\delta))}\right)\frac{\Phi_{\eta}(\mathcal{X})}{\pi_\Lambda},
\]

\[
S_2 = C\left(B + \sqrt{(\Phi_{\eta}(\mathcal{H})) + \log(1/\delta)}\right)\frac{\Phi_{\eta}(\mathcal{H})}{\pi_\Lambda}.
\]

We discuss about the bounds and setup of parameters before coming to the proof of Theorem 3.

**Remarks.** The regret bound in Theorem 3 enjoys best of both worlds from comparisons and direct queries. The first bound has the same form as in Theorem 2 but with another $2L_1\gamma + \zeta$ term. This comes from the first phase of COMP-GP-UCB, and the extra term comes from the fact that $f_c \neq f$. In the second phase, the algorithm achieves the second bound $S_2$, which is the rate of using $\pi_\Lambda$ direct queries to explore $\mathcal{H}_r$. Compared with $\Phi_{\eta}(\mathcal{X})$, the second bound has the same rate on $\pi_\Lambda$, but with a reduced search region $\mathcal{H}_r$ and a startup budget $\Lambda_0$ for comparisons to work. When $f_c$ is a good approximation to $f$, $\mathcal{H}_r$ is much smaller than $\mathcal{X}$ and will lead to a great improvement in the number of direct queries needed.

**Setup of parameters.** 1. Setting $\gamma$: $\gamma$ acts as a threshold for exploring comparisons in both phases of COMP-GP-UCB. A small $\gamma$ will lead to a small $\mathcal{H}_r$ and therefore better regret rates; but it will also make the algorithm spend more time on comparisons before moving to direct queries, i.e., a large $\Lambda_0$. One plausible choice for $\gamma$ is to set $\gamma = \frac{\zeta}{2L_2}$, and this will make $\mathcal{H}_r \approx \mathcal{H}$.

2. Setting $\beta_i$: The setup for $\beta_i$ in Theorem 3 requires knowing $\Phi_{\eta}(\mathcal{H}_r)$ before algorithm starts and this is unrealistic to set up. However, in practice the default choice for $\beta_i$ is often very loose and hand-tuned values are used instead (e.g., Kandasamy et al. (2016) uses $\beta_i = 0.2d\log(2t)$). In this sense this setup for $\beta_i$ does not affect its practical use. For theoretical purposes, we can also set $\beta_i = \beta_i^{(r)}$; this leads to a regret rate of $\tilde{O}\left(\frac{(B + \sqrt{\Phi_{\eta}(\mathcal{X})})\sqrt{\Phi_{\eta}(\mathcal{H}_r)}}{\sqrt{\pi_\Lambda}}\right)$, slightly larger than the current rate but still smaller than GP-UCB.

**Proof Sketch.** We prove Theorem 3 and Theorem 2 follows as a corollary. For the first bound, if we have left phase 1 and entered phase 2, let $T_0$ be the time that we leave phase 1. By routine calculation we can show

\[
S(\Lambda) \leq f^* - f(x_{T_0}) \leq L_1(f^*_r - f_r(x_{T_0})) \leq 2L_1\gamma + \zeta.
\]

On the other hand, if we do not finish phase 1 (e.g., when $\zeta = \gamma = 0$), we can follow the proof of IGP-UCB (Chowdhury and Gopalan, 2017) and show that

\[
S(\Lambda) \leq C\beta^{(r)}\left(\frac{\Phi_{\eta}(\mathcal{H}_r)}{\pi_\Lambda}\right) + \zeta.
\]

Combining (5) and (6) we get the first bound $S_1$. Now we show the second bound $S_2$. Suppose the algorithm makes $n$ queries. For any set $A \subseteq \mathcal{X}$, let $T_n^r(A)$ be the number of comparison queries into $A$ when the algorithm has made $n$ queries, and $T_n^d(A)$ be the number of direct queries. We have

\[
n = T_n^r(\mathcal{X}) + T_n^d(\mathcal{H}_r) + T_n^d(\mathcal{H}_r).
\]

For the first term, we show that there exists a constant $C_\kappa$ such that $T_n^r(\mathcal{X}) \leq C_\kappa \left(\frac{\beta_i}{\pi_\Lambda}\right)^{p+2}$, where $p = d$ for SE kernel and $p = 2d$ for Matérn kernel. For the second term, we show that our algorithm makes sure that $T_n^d(\mathcal{H}_r) = 0$, i.e., it always query in $\mathcal{H}_r$ when it uses direct queries. These two results combined can show that we allocate at least $\pi_\Lambda/2$ direct queries to explore $\mathcal{H}_r$. The second bound $S_2$ then follows by bounding the regret similar to IGP-UCB.

3.5 COMP-GP-UCB with Unknown $\zeta$

In practice, we cannot know $\zeta$ in general, and it is even hard to verify whether $\|f_c - f\|_\infty \leq \zeta$ holds for a given $\zeta$. On the other hand, we can often know an upper bound $\zeta_{\max}$ such that $\zeta \leq \zeta_{\max}$. For example, if we know both $f$ and $f_c$ are bounded in $[-B_{\infty}, B_{\infty}]$, we naturally have $\|f_c - f\|_\infty \leq 2B_{\infty}$. However, Algorithm 1 is not useful if we set $\zeta = 2B_{\infty}$, because that will lead to a constraint set $\mathcal{H} = \{x : f_c(x) \geq f^*_r - 2L_2B_{\infty}\}$ that can be as large as $\mathcal{X}$ and we have to explore the whole feature space with direct queries. We develop a slightly different method in
Algorithm 2 COMP-GP-UCB for unknown ζ

Input: Threshold γ, comparison bias starting point ζ₀, bias upper bound ζ_{max}, budget Λ
1: Set \( D₀ = D₀ = \emptyset, (μ₀(r), σ₀(r)) = (μ₀, σ₀) = (0, k^{1/2}), t \leftarrow 0, k \leftarrow 0, N_t \leftarrow 0 \)
2: repeat
3: \( x_t = \arg \max_{x \in \mathcal{X}} \mu_{t-1}(x) + \beta_t(r) \sigma_{t-1}(x) \)
4: \( \text{QUERY}(x_t, \text{comp}) \)
5: until \( \beta_t(r) \sigma_{t-1}(x) \leq \gamma \)
6: Let \( \hat{f}_t = \mu_{t-1}(x_t) - \beta_t(r) \sigma_{t-1}(x_t) \)
7: while \( \zeta_k \leq \zeta_{max} \) do
8: Let \( \phi_t(r) = \mu_{t-1}(x) + \beta_t(r) \sigma_{t-1}(x) - \hat{f}_t + 2L_2 \zeta_k \)
9: \( x_t = \arg \max_{x \in \mathcal{X}: \phi_t(r)(x) \geq \mu_{t-1}(x) + \beta_t(r) \sigma_{t-1}(x)} \)
10: if \( \beta_t(r)(x) \sigma_{t-1}(x) \geq \gamma \) then
11: \( \text{QUERY}(x_t, \text{comp}) \)
else
12: \( \text{QUERY}(x_t, \text{label}) \)
13: \( N_t \leftarrow N_t + 1 \)
14: if \( N_t \geq \frac{\Lambda}{\log(\log(\zeta_{max}/\zeta_0))} \) then
15: \( N_t \leftarrow 0, \zeta_{k+1} \leftarrow 2\zeta_k, k \leftarrow k + 1 \)
end if
18: \( t \leftarrow t + 1 \)
19: end while

Algorithm 2 that tries to search \( \zeta \) between an initial value \( \zeta_0 \) and the upper bound \( \zeta_{max} \), and adapts to the true \( \zeta \).

Algorithm 2 works in the finite-horizon scenario, where the budget Λ is given as input. The process of Algorithm 2 is mostly similar to Algorithm 1, except that it uses \( \zeta_k \) in the second phase in place of \( \zeta \). We optimize the function as if Assumption 1 holds for \( \zeta_k \). \( \zeta_k \) starts from \( \zeta_0 \) at step 13, once we have spent enough queries at the current estimate of \( \zeta \), we double the current \( \zeta_k \). We iterate until we reach \( \zeta_k > \zeta_{max} \). The threshold for \( N_t, \frac{\Lambda}{\log(\log(\zeta_{max}/\zeta_0))} \), is chosen such that we choose a label budget of \( \frac{\Lambda}{2} \) direct queries equally among the \( \log(\zeta_{max}/\zeta_0) \) possible values of the \( \zeta_k \)'s. The constant 2 is chosen arbitrarily here; any choice of \( \frac{\Lambda}{c} \) for a constant \( c > 1 \) will obtain the same rate.

We present our theoretical results as a corollary to Theorem 3. Since we cannot find the exact \( \zeta \), our results depend on a slightly larger \( \zeta = \max\{2\zeta_0\} \). We use \( \tilde{H} = \{ x \in \mathcal{X} : f(x) \geq f^* - 2L_2 \zeta - 4\} \) to represent the constraint set of interest when \( \zeta \) is replaced by \( \tilde{\zeta} \).

Corollary 4. Suppose Assumptions 2 and 1 hold, and \( \zeta \leq \zeta_{max} \). Under the same setting of \( \beta_t(r), c, A_0 \) as in Theo-
rem 3 and \( \beta_t = 2B + \sqrt{2 \left( \Phi_{t-1}(\tilde{H}) + 1 + \log(1/\delta) \right)} \), the simple regret of Algorithm 2 satisfies \( S(\Lambda) \leq \min\{S_1, S_2\} \), where
\begin{align*}
S_1 &= 2L_1 \gamma + 2\zeta + C \left( B + \sqrt{\Phi_{\pi_{\Lambda}}(\tilde{H}) + 1 + \log(1/\delta)} \right) \sqrt{\Phi_{\pi_{\Lambda}}(\tilde{H})}, \\
S_2 &= C \left( B + \sqrt{\Phi_{\pi_{\Lambda}}(\tilde{H}) + 1 + \log(1/\delta)} \right) \sqrt{\Phi_{\pi_{\Lambda}}(\tilde{H})}.
\end{align*}

Remark. 1. The regret rate of Corollary 4 is almost the same as Theorem 3 except the set \( \tilde{H} \) is replaced with \( H \). We note that all the terms in the regret rate depend only on \( \gamma \) or \( \zeta \), and do not depend on \( \zeta_{max} \). This means Algorithm 2 can adapt to unknown level of comparison bias \( \zeta \).
2. Similar to Theorem 3 Corollary 4 also requires the unknown quantity \( \Phi_k(x^\gamma) \) to set \( \beta_t \); in practice we can also use a similar hyper-parameter search to find this quantity. \( \gamma \) also takes a similar effect as in Algorithm 1 and \( \gamma = \frac{1}{2L_2} \zeta_0 \) can lead to \( H = H \) and a practical algorithm. Again, setup of these parameters only depends on \( \zeta \) and is not affected by \( \zeta_{max} \).

3.6 Comparison with MF-GP-UCB (Kandasamy et al., 2016)

Our setting and method share some common characteristics as the multi-fidelity method MF-GP-UCB (Kandasamy et al., 2016), and we formally discuss them here. Our setup is similar to MF-GP-UCB in the two-fidelity case, where the algorithm has access to the target function \( f \) and its approximation \( f^{(1)} \), with \( \|f - f^{(1)}\|_{\infty} \leq \zeta \) for some known \( \zeta > 0 \). Although we also assume \( f \) is a good approximation for \( f \) (in a weaker sense of being close in terms of \( f^* \) and \( f^{(1)} \), see Assumption 1), our setting is harder than MF-GP-UCB and their algorithm cannot be directly applied in our case. This is because we cannot directly query \( f \): \( f \) is only available through comparisons, and we will get the same set of comparisons from \( f \) and \( f + c \) for any constant \( c \). In our case, we can only get unbiased estimates for \( f_r \). However, it is unlikely that \( \|f_r - f\|_\infty \) is small, because \( f_r(x) \in [0, 1] \) for all \( x \) since it is the probability of beating a random point, whereas \( f \) can have arbitrary values.

MF-GP-UCB bears some resemblance to the second phase in our Algorithm 1, but they are principally different in choosing the next query point \( x_t \). In the MF-GP-UCB algorithm, we have access to another function \( f' \) similar to \( f \). The algorithm constructs two sets of UCBs \( \phi(x), \phi'(x) \) for \( f \) and \( f' \) separately, and use \( \min\{\phi(x), \phi'(x)\} \) as a final UCB. In our case, UCBs of \( f \) and \( f \) are not comparable. Instead we use a novel constrained optimization
approach based on observations in the first phase.

Another difference is that MF-GP-UCB needs the function difference \( \zeta \) known beforehand, whereas our modified COMP-GP-UCB (Algorithm 2) can adapt to an unknown \( \zeta \). We note that MF-GP-UCB does use a doubling mechanism in their experiments to make it practical, but they do not provide any theoretical guarantees.

4 Experiments

We perform experiments against plausible baselines to verify our theory and illustrate the efficacy of our algorithm, on both synthetic and real-world data. For comparison, we include state-of-art algorithms in the label-only and comparison-only setting, as well as an adapted version of MF-GP-UCB, as described below.

4.1 Experiment Setup

For synthetic data, we use functions from the multi-fidelity literature to produce comparisons on a lower fidelity and direct labels on a higher fidelity. In particular we use Currin exponential (CurrinExp, \( d = 2 \)) and Borehole (\( d = 8 \)) (Xiong et al. 2013) functions. We note that \( f \) and \( f_c \) have different values and maximizers. We do NOT add additional noise on direct queries for \( f \).

For real-world data, we experiment with the SVM tuning task from the MF-GP-UCB paper Kandasamy et al. (2016) and train a SVM classifier on the MAGIC Gamma dataset Dua and Graff (2017). We tune the RBF kernel bandwidth and the soft margin coefficient within range \((10^{-5}, 10^1)\) and \((10^{-3}, 10^5)\). We randomly sample a training set of size 2,000 and a validation set of 500 from the original dataset. Direct labels take in the specified bandwidth and margin coefficient and return the corresponding validation accuracy. On the other hand, comparisons only use a (fixed) subset of size 500 from the training set (noisy but cheap) and return the validation accuracy on the same validation data.

4.2 Baselines and Implementation Details.

Baselines. We evaluate the performance of COMP-GP-UCB against the following baselines: (1) GP-UCB (Srinivas et al. 2009): The label-only algorithm optimizing UCB of GP posterior. (2) KernelSelfSparring (Sui et al. 2017): A comparison-only algorithm that uses Thompson Sampling to optimize comparisons. We note that since \( f \neq f_c \), optimizing comparisons cannot lead to the global optimum. (3) MF-GP-UCB (Kandasamy et al. 2016): Although MF-GP-UCB is not directly applicable in our case, we try to use it by using comparisons as the lower fidelity. When the algorithm selects to query the lower fidelity on \( x_t \), we compare \( x \) to a random point \( X \in \mathcal{X} \) and use the result as feedback, the same process as COMP-GP-UCB.

Experiment Setup. We apply common techniques in Bayesian optimization to set up hyperparameters of each algorithm (we detail the implementation in appendix). All methods use the RBF kernel for GP. For all methods we compute the simple regret \( \mathbb{E}[R] \) w.r.t. \( f \). The results are averaged over 20 runs.

Cost Ratio. In practice, the relation between the costs of labels and comparisons can be complex. We call \( \frac{\lambda_c}{\lambda_l} \) the cost ratio between labels and comparisons; the larger the cost ratio, the cheaper the comparisons. Our algorithm generally works for a cost ratio \( \frac{\lambda_c}{\lambda_l} > 1 \). We test the performance under various cost ratios in our experiment. Unless otherwise specified, for all the experiments with a varying budget we set up the budget of MF-GP-UCB and use \( \lambda_c = 0.1 \) and \( \lambda_l = 1 \), and use a total budget of \( \Lambda = 100 \). For all the experiments with a varying cost ratio, we set \( \Lambda = 100, \lambda_l = 1 \), and vary the cost ratio between \([1, 10]\).

4.3 Results

Results on synthetic data. The results are summarized in Figure 1a & 1b. Firstly we compare the performance on CurrinExp by varying the total budget from 10 to 100 (Figure 1a). COMP-GP-UCB shows the best performance over all budget setups. It is worth noting that MF-GP-UCB performs worse than label-only GP-UCB in our setting; this is because the target function of MF-GP-UCB in this case essentially optimizes the function \( f_r \), which is bounded in \([0, 1]\), resulting in a very large approximation bias. In contrast, COMP-GP-UCB is able to use comparisons in an efficient way to reduce the search space for optimization. In the appendix, we also include the case where \( f = f_c \) and \( \lambda_c = \lambda_l \) (i.e., no bias on comparisons and cost ratio equals 1).

Then in Figure 1b we fix the total budget to be \( \Lambda = 100 \) and cost of labels \( \lambda_l = 1 \), and vary the cost ratio from 1 to 10 by varying comparison costs. COMP-GP-UCB achieves the best performance for all setups except for \( \lambda_c = \lambda_l = 1 \) when it is worse than the label-only GP-UCB algorithm. This is expected since our algorithm targets to use cheaper comparisons. Our algorithm can be more effective even with a fairly small cost ratio \( \geq 2 \).

Alternative Definitions of Regret. Since the simple regret notion defined for settings with both direct and comparison queries is necessarily unfair for methods designed to handle only one query type, we perform an additional experiment to examine the comparison regret and direct query regret separately. Namely, define

\[
R_c(A) = \min_{t, q_t = \text{comp}} f^* - f(x_t), \quad \text{if } \exists t \text{ s.t. } q_t = \text{comp},
\]

\[
\infty, \quad \text{otherwise.}
\]

Footnotes:

2 We find that KernelSelfSparring is extremely slow for \( d = 8 \) so we only test it for CurrinExp.

3 Due to space limits, we include results on the Borehole function, along with other experiment results in the appendix.
Figure 1: Empirical results comparing COMP-GP-UCB with baseline methods. KSS stands for KernelSelfSparring.

\[
R_t(\Lambda) = \begin{cases} 
\min_{t,q:t=q} f^* - f(x_t), & \text{if } \exists t \text{ s.t } q_t = \text{label}, \\
\infty, & \text{otherwise.}
\end{cases}
\]

In Figure 2 we plot the comparison regret \( R_c \) and direct query regret \( R_t \) for the CurrinExp synthetic function (same setting as Figure 1a). We plot \( R_c \) and \( R_t \) respectively for both COMP-GP-UCB and MF-GP-UCB, \( R_t \) for GP-UCB, and \( R_c \) for KSS. Our results show that even if we only consider comparisons or direct queries, COMP-GP-UCB still outperforms the baselines in most budget settings. Overall, COMP-GP-UCB has a large direct query regret initially because we mostly do comparisons in the initial stage, but it achieves a low regret when we have a larger budget. We note that COMP-GP-UCB can still query both comparisons and direct queries in this setting, and it leads to a better regret for both comparisons and direct queries.

Results on real data. The results are in Figure 1c & 1d. KernelSelfSparring (KSS) has an error rate much higher than other methods (larger than 0.16), so we exclude it in the plot. Similar to the synthetic data case, COMP-GP-UCB outperforms other baselines. The advantage of COMP-GP-UCB over GP-UCB is smaller than the synthetic case, which might possibly result from the larger difference between \( f_c \) and \( f \). Note that for real data we still vary the cost ratio because in practice the actual cost ratio might depend on various factors like computational cost and data collection. Nevertheless, Figure 1d shows that COMP-GP-UCB has an advantage over the baselines for cost ratio at least 2; this is very likely to happen since the training set for comparisons is only 1/4 the size of that for direct queries.

5 Conclusion

We consider a novel dueling-choice setting when both direct queries and comparisons are available for non-convex optimization. We propose the COMP-GP-UCB algorithm that can achieve benign regret rates in the dueling-choice setting, and can adapt to unknown biases in the comparisons. Our algorithm can also be of independent interest for other multi-fidelity or transfer learning settings where information gleaned from one fidelity or source domain can be actively transferred to optimize the target domain function, under milder conditions than existing literature.

Acknowledgements. This work has been supported in part by DARPA FA8750-17-2-0130, NSF CCF-1763734, and AFRL FA8750-17-2-0212.
A Review of the GP-UCB and IGP-UCB Algorithms

The GP-UCB (Srinivas et al., 2009) and IGP-UCB (Chowdhury and Gopalan, 2017) can be unified as in Algorithm 3. The algorithms only differ at their assumptions and thus the choice of $\beta_t$. Our setting of $\beta_t^{(r)}$ and $\beta_t$ is similar to IGP-UCB, as we focus more on the agnostic function setting.

Algorithm 3 GP-UCB and IGP-UCB

Input: Budget $\Lambda$

1. Set $D_0 = \emptyset$, $(\mu_0^{(l)}, \sigma_0^{(l)}) = (0, \kappa^{1/2})$, $t \leftarrow 0$
2. for $t = 1, 2, \ldots, \Lambda$ do
3. Compute $x_t = \arg \max_{x \in X} \mu_{t-1}(x) + \beta_t \sigma_{t-1}(x)$
4. Query $f(x_t)$ and obtain feedback $y_t$
5. Use $y_t$ and (2) to perform posterior updates, and obtain $\mu_t^{(l)}, \sigma_t^{(l)}$
6. end for

B Experiment Details

We apply basic techniques in Bayesian Optimization to conduct the experiments.

Initial queries: All the algorithms were initialized with uniform random queries with an initial budget of $\Lambda_0 = 10$. For multi-fidelity methods (MF-GP-UCB and COMP-GP-UCB), we use $\Lambda_0/2$ on comparisons and $\Lambda_0/2$ on direct queries; for GP-UCB we use all $\Lambda_0$ on labels.

Choice of kernel parameters: We estimate the kernel bandwidth and scale by maximizing marginal likelihood with respect to the initial random queries. We also update the kernel parameters by maximizing the marginal likelihood for the GP over the lower fidelity function after every 20 iterations, and for the GP over the true function after every 5 iterations.

Setup of $\beta_t$ and $\beta_t^{(r)}$: We follow MF-GP-UCB and set $\beta = 0.5 \times \log(0.5 \times t + 1) = \beta_t^{(r)}$.

Choosing query points. We use the DiRect algorithm (Jones et al., 1993) to maximize the marginal likelihood subject to parameter bounds, and to find the next query points.

Choice of hyperparameters $\zeta, \gamma$. For synthetic experiments we set $\zeta = f(x^*) - f_c(x^*)$, where $x^*$ is the maximizer of $f$. For experiments on real data, we heuristically set $\zeta$ to be a large enough value (we use 0.15 in our experiments). We initialize $\gamma = \text{range}(f) \cdot \zeta$, and whenever COMP-GP-UCB has queried 10 comparisons in a row, we double $\gamma$ (following MF-GP-UCB).

B.1 Additional Results on Borehole

The result on Borehole is depicted in Figure 3a. KSS is extremely slow for Borehole due to the dimension ($d = 8$), so we exclude it from the plot. As with CurrinExp, COMP-GP-UCB achieves the best performance with a large gap under
Figure 4: Empirical results comparing COMP-GP-UCB with baselines, under a single fidelity and $\lambda_c = \lambda_l = 1$. KSS stands for KernelSelfSparring.

all budget setups. Different than CurrinExp, COMP-GP-UCB displays an advantage over the baselines in all cost ratios including $\lambda_l = \lambda_c$ (Figure 3(b)). This might suggest that it is easier to find the maximum using $f_r$ (the lower fidelity) than using $f$ (the higher fidelity).

B.2 Results with Single Fidelity and Same Cost

Although COMP-GP-UCB is designed for the case where comparisons are cheaper than direct queries, it is also interesting to see how it performs when comparisons and direct queries cost the same. We conduct an experiment with $f = f_c$ and $\lambda_c = \lambda_l = 1$ in Figure 4. Note that MF-GP-UCB is not applicable in this setting since it is for the multi-fidelity setting. COMP-GP-UCB performs on par (for CurrinExp) or better (for Borehole) than the baselines in our result. Note that while our theory suggests that the convergence rate of COMP-GP-UCB is the same as GP-UCB when $\lambda_c = \lambda_l$ and $f = f_c$, practice the underlying function $f_r$ (of COMP-GP-UCB) might be easier to optimize than $f$, because it is bounded and can be smoother than $f$. For the case of CurrinExp, $f_r$ and $f$ are of similar shape and values, so COMP-GP-UCB leads to a slower convergence rate because comparisons are less informative than direct queries; for Borehole, the $f_r$ appears to be easier to optimize than the higher fidelity. One possible reason is that $f_r$ is smoother in shape than $f$ since it is bounded in $[0, 1]$; this interesting phenomenon needs further investigation as part of future work. We note that cost ratio $\frac{\lambda_l}{\lambda_c} = 2$ is enough for COMP-GP-UCB to surpass the performance of GP-UCB on CurrinExp (see Figure 1(b)).

C Proofs

C.1 Proof of Proposition 1

Let $x^*_c$ be a maximizer of $f_c$. Because of the link function assumption, $x^*_c$ is also a optimizer of $f_r$. We have

$$f^*_r - f_r(x) = \mathbb{E}[\sigma(f_c(x^*_c) - f_c(X))] - \mathbb{E}[\sigma(f_c(x) - f_c(X))]$$
$$= \mathbb{E}[\sigma(f_c(x^*_c) - f_c(X)) - \sigma(f_c(x) - f_c(X))]$$
$$\leq \mathbb{E}[L_2|f_c(x^*_c) - f_c(X)|] = L_2(f^*_c - f_c(x)).$$

The lower bound can be proved similarly.

C.2 Proof of Theorem 2 and 3

We show Theorem 3 and Theorem 2 follows as a direct corollary. We first use results in IGP-UCB (Chowdhury and Gopalan, 2017) to obtain confidence bands of $f_r$:

Lemma 5 (Theorem 2, Chowdhury and Gopalan, 2017). Define $\beta^{(r)}_t = 2||f_r||_c + \sqrt{2(\Phi_{t-1}(X) + 1 + \log(2/\delta))}$. 

C.3 Proof of Proposition 1
Then with probability $1 - \delta/2$ we have for all time $t$ and any point $x \in \mathcal{X}$,

$$|\mu_t^{(r)}(x) - f_r(x)| \leq \beta_t^{(r)} \sigma_{t-1}^{(r)}(x).$$

Lemma 5 also applies to $f, x \in \mathcal{H}_\gamma, \mu^{(l)}, \sigma^{(l)}$ by setting $\beta_t = 2\|f\|_\kappa + \sqrt{2(\Phi_{t-1}(\mathcal{H}^\gamma) + 1 + \log(1/\delta))}$.

We also use the following lemma to bound the sum of posterior variances:

**Lemma 6** (Lemma 8, Kandasamy et al. 2016). Let $A \subseteq \mathcal{X}$. Suppose we have $n$ queries $(x_i)_{i=1}^n$ of which $s$ points are in $A$. Then the posterior $\sigma_t$ satisfies

$$\sum_{x_t \in A} \sigma_{t-1}^2(x_t) \leq \frac{2}{\log(1 + \eta^{-2})} \Phi_s(A).$$

Suppose the event in Lemma 5 holds for $f$ and $f_r$. We first prove the first bound by looking at comparison queries. Firstly, in the first phase when we compute $x_t$ in step 3 we have

$$f_r^* - f_r(x_t) \leq \mu_{t-1}^{(r)}(x_t^*) + \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t^*) - (\mu_{t-1}(x_t) - \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t))$$

$$\leq \mu_{t-1}(x_t) + \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t) - (\mu_{t-1}^{(r)}(x_t) - \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t))$$

$$= 2\beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t). \quad (7)$$

The first inequality uses Lemma 5 and the second inequality is from that $x_t$ is the maximizer of $\mu_{t-1}^{(r)}(x) + \beta_t \sigma_{t-1}^{(r)}(x)$. Suppose we finish phase 1 and enter phase 2. Let $T_0$ be the time we leave phase 1, then we must have $\beta_{T_0} \sigma_{T_0-1}^{(r)}(x_{T_0}) \leq \gamma$. So

$$S(A) \leq f^* - f(x_{T_0})$$

$$\leq f_r^* - f_r(x_{T_0}) + \zeta$$

$$\leq f_r^* - f_r(x_{T_0}) + \zeta$$

$$\leq L_1(f_r^* - f_r(x_{T_0})) + \zeta \leq 2L_1 \gamma + \zeta. \quad (8)$$

The second inequality is from Assumption 1, the fourth inequality is from Assumption 2 and the last inequality is from (7).

If we do not finish phase 1, then the number of comparison queries is $N_1 \geq \pi_L - 1$ and $N_1 \leq \pi_L$, and we have

$$\left(\sum_{t} (f_r^* - f_r(x_t))\right)^2 \leq N_1 \sum_{t: q_t = \text{comparison}} (f_r^* - f_r(x_t))^2$$

$$\leq N_1 \sum_{t: q_t = \text{comparison}} 4 \left(\frac{\beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t)}{N_1}\right)^2$$

$$\leq C_1 N_1 \left(\frac{\beta_t^{(r)}}{N_1}\right)^2 \Phi_{N_1}(\mathcal{H}^\gamma). \quad (9)$$

Here $C_1 = \frac{8}{\log(1 + \eta^{-2})}$. The first step is from the Cauchy-Schwarz inequality, and the second step is from (7); the last step uses Lemma 6.
So
\[ S(\Lambda) \leq \frac{1}{N_1} \sum_t (\ell^* - f(x_t)) \]
\[ \leq \frac{2L_1}{N_1} \sum_t (\ell^*_r - f_r(x_t)) + \zeta \]
\[ \leq \frac{\beta N_1}{N_1} \sqrt{C_1N_1\Phi N_1(H)} + \zeta \]
\[ \leq \frac{C\beta}{\pi_L} \sqrt{\pi_L \Phi \pi_L(H)} + \zeta \]
\[ \leq C \left( B + \sqrt{\Phi \pi_L(H) + \log(1/\delta)} \right) \frac{\Phi \pi_L(H)}{\pi_L} + \zeta. \tag{10} \]

The second inequality follows from the same process as in (8); the third inequality follows from (9); the fourth inequality is from \( N_1 \geq \pi_L \geq \pi_L - 1 \). Here \( C \) is a constant whose value may change from line to line. Combining (8) and (10) we get the first bound.

To show the second bound, we examine the regret from direct queries. We first show that \( x^* \) is never excluded from our feasible region:

**Claim 7.** \( \phi_t^{(r)}(x^*) \geq 0 \) for all \( t \).

**Proof.** Suppose \( x^* \) is a maximizer of \( f \) in \( \mathcal{X} \). Then we have
\[ \phi_t^{(r)}(x^*) = \mu_t^{(r)}(x^*) + \beta_t^{(r)} \sigma_t^{(r)}(x^*) - \max_{x'} \left\{ \mu_t^{(r)}(x') - \beta_t^{(r)} \sigma_t^{(r)}(x') \right\} + L_2 \zeta \]
\[ \geq f_r(x^*) - f_r^* + L_2 \zeta \]
\[ = -L_2(f_r^* - f_r(x^*)) + L_2 \zeta \]
\[ \geq -L_2(f^* - f(x^*)) + L_2 \zeta \]
\[ = -L_2 \zeta + L_2 \zeta = 0. \]

The first inequality is from Lemma 5, the second inequality is from Assumption 1. \( \square \)

Let \( N \) be the (random) total number of queries under budget \( \Lambda \). We know that the support of \( N \) lies in \([n_L, \pi_L]\); we now suppose \( n \) is any number in \([n_L, \pi_L]\), and prove properties of Algorithm 1 when it uses \( n \) queries.

For any set \( A \subseteq \mathcal{X} \), let \( T_n^r(A) \) be the number of comparison queries into \( A \) when the algorithm has made \( n \) queries, and \( T_n^d(A) \) be the number of direct queries. We have
\[ n = T_n^r(\mathcal{X}) + T_n^d(H) + T_n^d(\overline{H}) \]
since \( \overline{H} \cup H = \mathcal{X} \). We bound the first two terms using the following two lemmas:

**Lemma 8.** There exists a constant \( C_\kappa \) dependent on \( \kappa, d \) such that \( T_n^r(\mathcal{X}) \leq C_\kappa \left( \frac{\beta^{(r)} \gamma^{(r)}}{\pi_L} \right)^{p+2} \), where \( p = d \) for SE kernel and \( p = 2d \) for Matérn kernel.

This lemma is proved in Section C.3.

**Lemma 9.** \( T_n^d(\overline{H}) = 0 \).

**Proof.** Suppose \( q_t \) = label for some \( t \). Then we must have \( \phi_t^{(r)}(x_t) \geq 0 \), and that \( \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t) < \gamma, \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t^{(r)}) < \)
\[ f_c(x_t) \geq \mu_{t-1}^{(r)}(x_t) - \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t) \]

\[ = \phi_t^{(r)}(x_t) - 2\beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t) + \max_{x'} \left\{ \mu_{t-1}^{(r)}(x') - \beta_t^{(r)} \sigma_{t-1}^{(r)}(x') \right\} - L_2 \zeta \]

\[ \geq \phi_t^{(r)}(x_t) - 2\beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t) + \mu_{t-1}^{(r)}(x_t) - \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t) - L_2 \zeta \]

\[ \geq 0 - 2\gamma + f^*_r - 2\gamma - L_2 \zeta \]

\[ = f^*_r - 4\gamma - L_2 \zeta. \]

The first inequality is by applying Lemma 5, the second inequality is by letting \( x' = x_t^{(r)} \); the third inequality is by noticing that

\[ \mu_{t-1}^{(r)}(x_t^{(r)}) - \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t^{(r)}) \geq \left[ \mu_{t-1}^{(r)}(x_t^{(r)}) + \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t^{(r)}) \right] - 2\beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t^{(r)}) \geq f^*_r - 2\gamma. \]

Lemma 8 shows that we will not make too many queries on comparisons, whereas Lemma 9 shows that we always query \( x_t \in \mathcal{H}^c \) when \( q_t = \text{label} \). Now let \( N_0 = \text{the smallest number such that for any } \Lambda \geq N_0 \lambda_c \), we have

\[ \lambda_{t} \left( \frac{2r \sqrt{d}}{\varepsilon \pi_{\Lambda}} \right)^{d} \left( \frac{2\eta \beta^{(r)}_{\pi_{\Lambda}}}{\gamma} \right)^{2} C_{\kappa} \left( \frac{\beta^{(r)}_{\pi_{\Lambda}}}{\gamma} \right)^{p+2} \leq \frac{\Lambda}{2} \]

where \( C_{\kappa} \) and \( p \) are defined in (12). Such \( \Lambda_0 \) is guaranteed to exist since \( \beta^{(r)}_{\pi_{\Lambda}} \) grows in sublinear rate for linear, SE and Matérn kernels on \( \pi_{\Lambda} \), and therefore \( \Lambda \). Thus the number of queries to \( f_c, T^*_n(\mathcal{X}) \), is at most \( \frac{n}{2} \lambda_c^{2} \), and therefore we query at least \( \frac{n}{2} \lambda_c^{2} \) times on direct queries, \( T^*_n(\mathcal{H}^c) \geq \frac{n}{2} \lambda_c^{2} \).

We now follow a similar path as for comparison queries to bound the regret based on direct queries. Note that if we use direct query at round \( t \), we have \( x_t \in \mathcal{H}^c \) and that

\[ f^* - f(x_t) \leq \mu_{t-1}^{(l)}(x^*) + \beta_t \sigma_{t-1}^{(l)}(x^*) - (\mu_{t-1}^{(l)}(x_t) - \beta_t \sigma_{t-1}^{(l)}(x_t)) \]

\[ \leq \mu_{t-1}^{(l)}(x_t) + \beta_t \sigma_{t-1}^{(l)}(x_t) - (\mu_{t-1}^{(l)}(x_t) - \beta_t \sigma_{t-1}^{(l)}(x_t)) \]

\[ = 2\beta_t \sigma_{t-1}^{(l)}(x_t). \]

The first inequality uses Lemma 5 and the second inequality uses Claim 7 and that \( x_t \) is the maximizer of \( \mu_{t-1}^{(l)}(x) + \beta_t \sigma_{t-1}^{(l)}(x) \).

Now therefore

\[ \left( \sum_{t: q_t = \text{label}} (f^* - f(x_t)) \right)^2 \leq T^*_n(\mathcal{H}^c) \sum_{t: q_t = \text{label}} (f^* - f(x_t))^2 \]

\[ \leq T^*_n(\mathcal{H}^c) \sum_{t: q_t = \text{label}} 4 \left( \beta_t \sigma_{t-1}^{(l)}(x_t) \right)^2 \]

\[ \leq C_1 T^*_n(\mathcal{H}^c) (\beta_n)^2 \Phi_{T^*_n(\mathcal{H}^c)}(\mathcal{H}^c) \]

Here \( C_1 = \frac{8}{\log(1+\eta+n)} \). The first step is from the Cauchy-Schwarz inequality, and that \( T^*_n(\mathcal{H}^c) = 0 \); the second step is from (11); the last step uses Lemma 5.
We use the following lemma from (Kandasamy et al., 2016)

\( S(\Lambda) \leq \frac{1}{T_n(\mathcal{H}^{\gamma})} \sum_{x \in \mathcal{H}^{\gamma}} (f^* - f(x)) \)

\[ \leq \frac{\beta_N}{T_n(\mathcal{H}^{\gamma})} \sqrt{C_1 T_n(\mathcal{H}^{\gamma}) \Phi_{T_n(\mathcal{H}^{\gamma})} (\mathcal{H}^{\gamma})} \]

\[ \leq \frac{\beta_N}{n} \sqrt{\Phi_{nA}(\mathcal{H}^{\gamma})(\mathcal{H}^{\gamma})} \]

\[ \leq C \left( B + \sqrt{\left( \Phi_{nA}(\mathcal{H}^{\gamma}) + \log(1/\delta) \right)} \right) \sqrt{\frac{\Phi_{nA}(\mathcal{H}^{\gamma})}{n}}. \]

C.3 Proof of Lemma 8

We use the following lemma from (Kandasamy et al., 2016).

**Lemma 10** (Lemma 13, (Kandasamy et al., 2016)). Let \( A \subseteq \mathcal{X} \) such that its L2 diameter \( \text{diam}(A) \leq D \). Say we have \( n \) queries \((x_t)_{t=1}^n\) of which \( s \) points are in \( A \). Then the posterior variance of the GP, \( \kappa'(x, x) \) at any \( x \in A \) satisfies

\[ \kappa'(x, x) \leq \begin{cases} C_{SE}D^2 + \frac{n^2}{\gamma^2}, & \text{if } \kappa \text{ is the SE kernel,} \\ C_{Mat}D + \frac{n^2}{\gamma^2}, & \text{if } \kappa \text{ is the Matérn kernel}, \end{cases} \]

for appropriate kernel dependent constants \( C_{SE}, C_{Mat} \).

Consider the SE kernel and the comparison oracle, and a \( \varepsilon_n = \frac{\beta_n^{(r)} \gamma \sqrt{C_{SE}}}{2} \) covering \((B_t)_{t=1}^n\) of \( \mathcal{X} \). We claim that the number of comparison queries inside any \( B_t \) is at most \( 2\left(\frac{4\beta_n^{(r)}}{\gamma} \right)^2\): suppose we have already queried \( 2\left(\frac{4\beta_n^{(r)}}{\gamma} \right)^2\) samples in \( B_t \) at some time \( t < n \). By Lemma 10 we have

\[ \max_{x \in B_t} \kappa^{(r)}_{t-1}(x, x) \leq C_{SE}(2\varepsilon_n)^2 + \frac{n^2}{\gamma^2} \leq \left( \frac{\gamma}{\beta_n^{(r)}} \right)^2. \]

Therefore \( \beta_n^{(r)} \sigma_{t-1}(x, x) \leq \beta_n^{(r)} \sigma_{t-1}(x, x) \leq \gamma. \) Note that whenever \( q_t = \text{comp} \), we always have \( \beta_n^{(r)} \sigma_{t-1}(x, x) \leq \gamma; \) so the event that \( q_t = \text{comp} \) and \( x_t \in B_t \) will not happen until time \( n \). We can obtain a similar result for Matérn kernel with \( \varepsilon_n = \frac{\gamma^2}{4C_{Mat}(\beta_n^{(r)})^2} \).

Therefore we have

\[ T_n^{(r)}(\mathcal{X}) \leq \Omega_{\varepsilon_n}(\mathcal{X})\left[ 2\left(\frac{4\beta_n^{(r)}}{\gamma} \right)^2 \right] \leq \left( \frac{2\sqrt{d}}{\varepsilon_n} \right)^d \left( \frac{2\eta \beta_n^{(r)}}{\gamma} \right)^2 = C_{\kappa} \left( \frac{\beta_n^{(r)}}{\gamma} \right)^{p+2}. \]

Here \( \Omega_{\varepsilon_n}(\mathcal{X}) \) is the covering number of \( \mathcal{X} \), and we bound the covering number as \( \Omega_{\varepsilon_n}(\mathcal{X}) \leq \left( \frac{2\sqrt{d}}{\varepsilon_n} \right)^d \). Here \( C_{\kappa} = 2^{2d+2r^d}d^dC_{SE}^d\eta^2 \) and \( p = d \) for SE kernel, while \( C_{\kappa} = 2^{3d+2r^d}d^dC_{Mat}^d\eta^2 \) and \( p = 2d \) for Matérn kernel.

C.4 Proof of Corollary 4

**Proof.** Firstly, for the first regret bound, we have the same guarantee as in Theorem 3 since the first phase is exactly the same. For the second bound, when \( \Lambda \geq \Lambda_0 \), we allocate at least budget of \( \Lambda/2 \) on direct queries. Since we double \( \zeta_k \) in each iteration, at some iteration \( k_0 = O(\log(\zeta/\zeta_0)) \) we will have \( \zeta_k \in [\zeta, \tilde{\zeta}] \). Let \( \bar{N} = \frac{n}{2\log(\zeta/\zeta_0)} \). From the proof of Theorem 3 we have the regret \( S(\Lambda) \leq \beta_N \sqrt{\frac{8\Phi_{\tilde{b}_N}(\mathcal{H}^{\gamma})}{\log(1+\eta^{-2})}N} \) in iteration \( k_0 \). The theorem then follows by realizing \( \beta_N, \Phi_{\tilde{b}_N}(\mathcal{H}^{\gamma}) \) grows sublinearly in \( n \).

---

4The original lemma from (Kandasamy et al., 2016) assumes \( f \sim GP(0, \kappa) \) and \( \varepsilon \sim \mathcal{N}(0, \eta^2) \), but exactly the same proof applies without these assumptions.