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 Φ 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, Φ 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 Kaelbling, 2000) and material science(Xue et al., 2016). With an unknown black-box function $f : \mathcal{X} \to \mathbb{R}$, 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-

Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI), PMLR volume 124, 2020.

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 precisions. 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 (Jamieson 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_r 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 $f : \mathcal{X} \to \mathbb{R}$, 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[1].

Comparison Probabilities. In addition to traditional direct queries y, we can obtain (cheap) comparisons for a pair of points $(x_t, x'_t) \in \mathcal{X} \times \mathcal{X}$. 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 $\Pr[x \succ x'] = \sigma(f_c(x) - f_c(x'))$ for some function σ. 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.

[1] Our methods can also be extended to the setting where ε follows a sub-Gaussian distribution with parameter η. We assume a bounded ε for simplicity here.

- Dueling-Choice Bandits with misspecified comparisons: We assume $f_c \approx f$. 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_c. To this end, we assume a bounded difference near the optimum:

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

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

  We note that our results can be generalized to the case where Assumption 1 only holds for $x \in \{x : f^* - f(x) \leq \tau\}$ for some fixed constant τ.

Smoothness Assumptions. We assume that the target function f lies in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}_\kappa$ induced by kernel κ, and that the RKHS norm of f is bounded: $\|f\|_\kappa \leq B$ for a known constant B. This assumption is also analyzed in (Chowdhury and Gopalan, 2017; Srinivas et al., 2009) for traditional bandits. We note that every function $f \in \mathcal{H}_\kappa$ has a finite kernel norm. When κ is the linear kernel, $\|f\|_\kappa \leq B$ induces that f is B-Lipschitz.

Budgets and Regrets. We analyze the problem of optimizing f under a given cost budget Λ. Suppose a direct query costs λ_l units of some resource and a comparison costs $\lambda_c < \lambda_l$. Also, let $\overline{n}_\Lambda = \lceil \frac{\Lambda}{\lambda_c} \rceil$ be the upper bound on number of queries when we use all the budget on comparisons, and $\underline{n}_\Lambda = \lfloor \frac{\Lambda}{\lambda_l} \rfloor$ be the corresponding lower bound when we only use direct queries. Also let $q_t =$ label if we make direct queries at iteration t, and $q_t =$ comp otherwise. We analyze the simple regret under budget Λ, defined as follows:

$$S(\Lambda) = \min_t r_t \tag{1}$$
$$= \min_t \begin{cases} f^* - f(x_t) & \text{if } q_t = \text{label}, \\ \min\{f^* - f(x_t), f^* - f(x_t')\}, & \text{if } q_t = \text{comp}. \end{cases}$$

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 $\tilde{O}(\cdot)$ 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 ζ 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 κ. 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., $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^t$ with $y_i = f(x_i) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \eta^2)$, then the posterior distribution at $f(x)|\mathcal{D}$ would be a Gaussian $\mathcal{N}(\mu_t(x), \sigma_t(x))$ with

$$\mu_t(x) = k^T (K + \eta^2 I_t)^{-1} Y, \tag{2}$$
$$\sigma_t(x) = \kappa(x, x) - k^T (K + \eta^2 I_t)^{-1} k.$$

Here $Y = (y_1, ..., y_t)^T$, $k = (\kappa(x, x_1), ..., \kappa(x, x_t))^T$, and matrix $K \in \mathbb{R}^{t \times t}$ is given by $K_{ij} = \kappa(x_i, x_j)$, and I_t is the $t \times t$ 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 feedbacks 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 ε 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; Srinivas et al., 2009).

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 $\tilde{A} = \{x_1, ..., x_n\} \subseteq A$ is a finite subset of A. Then the maximum information gain on A with n evaluations is defined as $\Phi_n(A) = \max_{\tilde{A} \subseteq A, |\tilde{A}| = n} I(f_{\tilde{A}} + \varepsilon_{\tilde{A}}; f_{\tilde{A}})$, where $f_{\tilde{A}} = [f(x)]_{x \in \tilde{A}}$, $\varepsilon_{\tilde{A}} \sim \mathcal{N}(0, \eta^2 I)$, and $I(X, Y) = H(X) - H(X|Y)$ 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 κ, $\Phi_n(\mathcal{X}) = O(d \log n)$; ii) for squared exponential (SE, or RBF) kernel, $\Phi_n(\mathcal{X}) = O((\log n)^{d+1})$; iii) For Matérn kernels $\kappa(x, x') = \frac{2^{1-\nu}}{\Gamma(\nu)} (\frac{\sqrt{2\nu}z}{\rho})^\nu B_\nu (\frac{\sqrt{2\nu}z}{\rho})$, we have $\Phi_n(\mathcal{X}) = O\left(n^{\frac{d(d+1)}{2\nu + d(d+1)}} \log n\right)$.

Review of GP-UCB and IGP-UCB. Previous sequential optimization has adopted the upper confidence bound

(UCB) principle, where we maintain a high-confidence upper bound $\phi : \mathcal{X} \to \mathbb{R}$ for all $x \in \mathcal{X}$, such that $f(x) \le \phi(x)$ with high probability. Our algorithm builds on UCB algorithms for GP, namely GP-UCB (Srinivas et al., 2009) and IGP-UCB (Chowdhury and Gopalan, 2017) (the latter is an improvement of the former). In time step t of optimization, IGP-UCB queries the point that maximizes the confidence upper bound in the form $\mu^{(l)}_{t-1}(x) + \beta_t \sigma^{(l)}_{t-1}(x)$, where $\mu^{(l)}_{t-1}, (\sigma^{(l)}_{t-1})^2$ are the posterior mean and variance function of the GP from step $t-1$, and β_t is a multiplier that increases with t. We describe these algorithms in detail in Appendix.

3.2 The Borda Function f_r

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 $x_1 \succ x_2 \succ x_3 \succ x_1$ (Zoghi et al., 2015).

We instead consider a different function directly related to f_c, defined as $f_r(x) = \Pr[x \succ X]$, where X is randomly chosen from $\mathcal{X}$. In words, $f_r(x)$ is the probability that x beats a random point $X \in \mathcal{X}$. We refer to f_r 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_r is that we can obtain unbiased estimates of $f_r(x)$ by comparing x to a random point in $X \in \mathcal{X}$.

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

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

In other words, difference in f_c will cause a difference of similar scale in f_r. This requires that the comparisons induces a Borda function f_r such that f_r is close to f_c at its optimum, and that f_r and f_c has 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 $f_c(x) - f_c(x')$, i.e., comparisons do not need to be consistent. We can show that Assumption 2 holds under the link function setting, when σ is Lipschitz continuous:

Proposition 1. *Suppose comparisons follows a link function σ 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 $\sigma(x) = \frac{1+x}{2}$ and cannot be applied for BTL or Thurstone models.

Algorithm 1 COMP-GP-UCB

Input: Comparison bias ζ, comparison exploration threshold γ, confidence δ

Set $D^r_0 = D^l_0 = \emptyset$, $(\mu^{(r)}_0, \sigma^{(r)}_0) = (\mu^{(l)}_0, \sigma^{(l)}_0) = (0, \kappa^{1/2})$, $t \leftarrow 0$
repeat
  Compute $x_t = \arg\max_{x \in \mathcal{X}} \mu^{(r)}_{t-1}(x) + \beta^{(r)}_t \sigma^{(r)}_{t-1}(x)$
  QUERY(x_t, comp)
until $\beta^{(r)}_t \sigma^{(r)}_{t-1}(x_t) \le \gamma$ or budget exhausted
Let $\hat{f}_r = \mu^{(r)}_{t-1}(x_t) - \beta^{(r)}_t \sigma^{(r)}_{t-1}(x_t)$
while Budget not exhausted **do**
  Let $\phi^{(r)}_t(x) = \mu^{(r)}_{t-1}(x) + \beta^{(r)}_t \sigma^{(r)}_{t-1}(x) - \hat{f}_r + L_2\zeta$
  Compute $x_t = \arg\max_{x \in \mathcal{X}: \phi^{(r)}_t(x) \ge 0} \mu^{(l)}_{t-1}(x) + \beta_t \sigma^{(l)}_{t-1}(x)$
  if $\beta^{(r)}_t(x_t)\sigma^{(r)}_{t-1}(x_t) \ge \gamma$ **then** QUERY(x_t, comp)
  else QUERY(x_t, label)
  end if
  $t \leftarrow t + 1$
end while

procedure QUERY(query point x_t, query type q_t)
  if $q_t =$ comp **then**
    Sample x' randomly from $\mathcal{X}$ and query to compare (x_t, x'), obtain z_t
    Update $D^c_t \leftarrow D^c_{t-1} \cup \{(x_t, z_t)\}$, $D^l_t \leftarrow D^l_{t-1}$
    Perform Bayesian update for $\mu^{(r)}_t, \sigma^{(r)}_t$ based on D^c_t with $y_t = z_t$ following (2)
  else
    Query direct labels for x_t and obtain y_t
    Update $D^l_t \leftarrow D^l_{t-1} \cup \{(x_t, y_t)\}$, $D^c_t \leftarrow D^c_{t-1}$
    Perform Bayesian update for $\mu^{(l)}_t, \sigma^{(l)}_t$ based on D^l_t following (2)
  end if
end procedure

3.3 Optimization with Known ζ

When ζ 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 Λ 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 $\mathcal{H} := \{x : f_r(x) \geq f_r^* - L_2\zeta\}$ where $f_r^* = \max_x f_r(x)$. $\mathcal{H}$ is guaranteed to contain the optimizer f under our assumptions; To see this, let x^* be any optimizer of f, and we have $f_r^* - f_r(x^*) \leq L_2(f_c^* - f_c(x^*)) \leq L_2(f^* - f(x^*) + \zeta) = L_2\zeta$. 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 $\mathcal{X}$ if comparisons are mostly correct (i.e., ζ is small); therefore we can explore more efficiently by restricting the search on $\mathcal{H}$.

COMP-GP-UCB takes as input ζ, a parameter γ to control exploration on comparisons, and a confidence level δ. We keep track of posteriors $(\mu^{(l)}, \sigma^{(l)}), (\mu^{(r)}, \sigma^{(r)})$ for f and f_r respectively, and construct confidence intervals $\mu_{t-1}^{(l)}(x) \pm \beta_t \sigma_{t-1}^{(l)}(x)$, $\mu_{t-1}^{(r)}(x) \pm \beta_t^{(r)} \sigma_{t-1}^{(r)}(x)$. Since f_r is unknown, to approximate $\mathcal{H}$, the algorithm adopts a two-phase approach: In the first phase (Step 2-5), we optimize f_r using comparison queries until $\beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t) \leq \gamma$, i.e., the queried point has confidence of at least γ. At the end of the first phase, we compute $\hat{f}_r$ as a lower bound for f_r^*. Next, we start the second phase exploring f (Step 7-14). We select the query point x_t based on a filtering $\phi_t^{(r)}(x) \geq 0$; the filtering approximates the constraint set $\mathcal{H}$ by combining the current UCB of f_r and the LCB $\hat{f}_r$ from the first phase. Then the algorithm optimizes the UCB of f under the constraint of $\phi_t^{(r)}(x) \geq 0$. 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 $\hat{f}_r \leq f_r^*$ 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 $\phi_t^{(r)}$. The choice of $\phi_t^{(r)}$ is critical for the algorithm to succeed. We want that the region $S = \{x : \phi_t^{(r)}(x) \geq 0\}$ 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_c and f. To achieve the latter, we need both a good UCB of f_r and a good LCB of $f_r^* = \max f_r(x)$. 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 $\hat{f}_r$; without the first phase $\hat{f}_r$ can be arbitrarily bad and it will lead to suboptimal direct queries. In the proof we show that when $\phi_t^{(r)}(x) \geq 0$ and $\beta_t^{(r)}(x)\sigma_{t-1}^{(r)}(x) \geq \gamma$, 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 $f_c = f$. Let $\beta_t^{(r)} = 2B + \sqrt{2\left(\Phi_{t-1}(\mathcal{X}) + 1 + \log(1/\delta)\right)}$. There exists a constant C dependent on d, κ such that COMP-GP-UCB with $\zeta = \gamma = 0$ has a simple regret bounded by*

$$S(\Lambda) \leq C\left(B + \sqrt{(\Phi_{\overline{n}_\Lambda}(\mathcal{X}) + \log(1/\delta))}\right)\sqrt{\frac{\Phi_{\overline{n}_\Lambda}(\mathcal{X})}{\overline{n}_\Lambda}}. \tag{3}$$

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

$$S_{\text{IGP-UCB}}(\Lambda) \leq \tag{4}$$
$$C\left(B + \sqrt{\left(\Phi_{\underline{n}_\Lambda}(\mathcal{X}) + \log(1/\delta)\right)}\right)\sqrt{\frac{\Phi_{\underline{n}_\Lambda}(\mathcal{X})}{\underline{n}_\Lambda}},$$

where $\underline{n}_\Lambda = \lfloor \frac{\Lambda}{\lambda_l} \rfloor$. This is the same form as (3), but with $\overline{n}_\Lambda$ replaced with $\underline{n}_\Lambda$. Recall that $\overline{n}_\Lambda$ is the number of queries when we use all the budget on comparisons, and $\underline{n}_\Lambda$ 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 ζ), the set $\mathcal{H} = \{x : f_r(x) \geq f_r^* - L_2\zeta\}$ is much smaller than the feature space $\mathcal{X}$. Therefore by using comparisons, we wish to replace the $\Phi_{n_0}(\mathcal{X})$ term in (4) by $\Phi_{n_0}(\mathcal{H})$, 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 γ, defined as $\mathcal{H}^\gamma = \{x \in \mathcal{X} : f_r(x) \geq f_r^* - L_2\zeta - 4\gamma\}$. The following theorem characterizes the regret of COMP-GP-UCB under the misspecified comparison setting.

Theorem 3. *Suppose Assumptions 2 and 1 hold, and ζ is known. Let $\beta_t^{(r)}$ be the same as in Theorem 2, and $\beta_t = 2B + \sqrt{2\left(\Phi_{t-1}(\mathcal{H}^\gamma) + 1 + \log(1/\delta)\right)}$. There exists constants Λ_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*

$$S_1 = 2L_1\gamma + \zeta + C\left(B + \sqrt{(\Phi_{\overline{n}_\Lambda}(\mathcal{X}) + \log(1/\delta))}\right)\sqrt{\frac{\Phi_{\overline{n}_\Lambda}(\mathcal{X})}{\overline{n}_\Lambda}},$$

$$S_2 = C\left(B + \sqrt{\left(\Phi_{\underline{n}_\Lambda}(\mathcal{H}^\gamma) + \log(1/\delta)\right)}\right)\sqrt{\frac{\Phi_{\underline{n}_\Lambda}(\mathcal{H}^\gamma)}{\underline{n}_\Lambda}}.$$

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 $\underline{n}_\Lambda$ direct queries to explore $\mathcal{H}^\gamma$. Compared with (4), the second bound has the same rate on $\underline{n}_\Lambda$, but with a reduced search region $\mathcal{H}^\gamma$ and a startup budget Λ_0 for comparisons to work. When f_c is a good approximation to f, $\mathcal{H}^\gamma$ 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 γ: γ acts as a threshold for exploring comparisons in both phases of COMP-GP-UCB. A small γ will lead to a small $\mathcal{H}^\gamma$ 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 Λ_0. One plausible choice for γ is to set $\gamma = \frac{1}{L_2}\zeta$, and this will make $\mathcal{H}^\gamma \approx \mathcal{H}$.
2. Setting β_t: The setup for β_t in Theorem 3 requires knowing $\Phi_t(\mathcal{H}^\gamma)$ before algorithm starts and this is unrealistic to set up. However, in practice the default choice for β_t is often very loose and hand-tuned values are used instead (e.g., Kandasamy et. al(Kandasamy et al., 2016) uses $\beta_t = 0.2d\log(2t)$). In this sense this setup for β_t does not affect its practical use. For theoretical purposes, we can also set $\beta_t = \beta_t^{(r)}$; this leads to a regret rate of $\tilde{O}\left(\frac{(B+\sqrt{\Phi_{\underline{n}_\Lambda}(\mathcal{X})})\sqrt{\Phi_{\underline{n}_\Lambda}(\mathcal{H}^\gamma)}}{\sqrt{\underline{n}_\Lambda}}\right)$, 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

$$S(\Lambda) \leq f^* - f(x_{T_0}) \leq L_1(f_r^* - f_r(x_{T_0})) \leq 2L_1\gamma + \zeta. \tag{5}$$

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

$$S(\Lambda) \leq C\beta_{\overline{n}_\Lambda}^{(r)}\sqrt{\frac{\Phi_{\overline{n}_\Lambda}(\overline{\mathcal{H}^\gamma})}{\overline{n}_\Lambda}} + \zeta. \tag{6}$$

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 $T_n^l(A)$ be the number of direct queries. We have

$$n = T_n^r(\mathcal{X}) + T_n^l(\overline{\mathcal{H}^\gamma}) + T_n^l(\mathcal{H}^\gamma).$$

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

3.5 COMP-GP-UCB with Unknown ζ

In practice, we cannot know ζ in general, and it is even hard to verify whether $\|f_c - f\|_\infty \leq \zeta$ holds for a given ζ. 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 $\mathcal{H} = \{x : f_r(x) \geq f_r^* - 2L_2B_\infty\}$ 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 ζ_0, bias upper bound $\zeta_{\max}$, budget Λ

1: Set $D_0^r = D_0^l = \emptyset$, $(\mu_0^{(r)}, \sigma_0^{(r)}) = (\mu_0^{(l)}, \sigma_0^{(l)}) = (0, \kappa^{1/2}), t \leftarrow 0, k \leftarrow 0, N_l \leftarrow 0$
2: **repeat**
3: Compute $x_t = \arg\max_{x \in \mathcal{X}} \mu_{t-1}^{(r)}(x) + \beta_t^{(r)} \sigma_{t-1}^{(r)}(x)$
4: QUERY(x_t, comp)
5: **until** $\beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t) \leq \gamma$
6: Let $\hat{f}_r = \mu_{t-1}^{(r)}(x_t) - \beta_t^{(r)} \sigma_{t-1}^{(r)}(x_t)$
7: **while** $\zeta_k \leq \zeta_{\max}$ **do**
8: Let $\phi_t^{(r)}(x) = \mu_{t-1}^{(r)}(x) + \beta_t^{(r)} \sigma_{t-1}^{(r)}(x) - \hat{f}_r + 2L_2\zeta_k$
9: Compute $x_t = \arg\max_{x \in \mathcal{X}: \phi_t^{(r)}(x) \geq 0} \mu_{t-1}^{(l)}(x) + \beta_t \sigma_{t-1}^{(l)}(x)$
10: **if** $\beta_t^{(r)}(x_t) \sigma_{t-1}^{(r)}(x_t) \geq \gamma$ **then** QUERY(x_t, comp)
11: **else**
12: QUERY(x_t, label)
13: $N_l \leftarrow N_l + 1$
14: **end if**
15: **if** $N_l \geq \frac{\underline{n}_\Lambda}{2\lceil \log(\zeta_{\max}/\zeta_0) \rceil}$ **then**
16: $N_l \leftarrow 0, \zeta_{k+1} \leftarrow 2\zeta_k, k \leftarrow k+1$
17: **end if**
18: $t \leftarrow t+1$
19: **end while**

Algorithm 2 that tries to search ζ between an initial value ζ_0 and the upper bound $\zeta_{\max}$, and adapts to the true ζ.

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 ζ_k in the second phase in place of ζ. We optimize the function as if Assumption 1 holds for ζ_k. ζ_k starts from ζ_0; at step 15, once we have spent enough queries at the current estimate of ζ, we double the current ζ_k. We iterate until we reach $\zeta_k > \zeta_{\max}$. The threshold for N_l, $\frac{\underline{n}_\Lambda}{2\lceil \log(\zeta_{\max}/\zeta_0) \rceil}$, is chosen such that we divide a label budget of $\underline{n}_\Lambda/2$ direct queries equally among the $\lceil \log(\zeta_{\max}/\zeta_0) \rceil$ possible values of the ζ_k's. The constant 2 is chosen arbitrarily here; any choice of $\underline{n}_\Lambda/c$ 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 ζ, our results depend on a slightly larger $\bar{\zeta} = \max\{2\zeta, \zeta_0\}$. We use $\hat{H}^\gamma = \{x \in \mathcal{X} : f_r(x) \geq f_r^* - 2L_2\bar{\zeta} - 4\gamma\}$ to represent the constraint set of interest when ζ is replaced by $\bar{\zeta}$.

Corollary 4. *Suppose Assumptions 2 and 1 hold, and $\zeta \leq \zeta_{\max}$. Under the same setting of $\beta_t^{(r)}, C, \Lambda_0$ as in Theorem 3, and $\beta_t = 2B + \sqrt{2\left(\Phi_{t-1}(\hat{H}^\gamma) + 1 + \log(1/\delta)\right)}$, the simple regret of Algorithm 2 satisfies $S(\Lambda) \leq \min\{S_1, S_2\}$, where*

$$S_1 = 2L_1\gamma + 2\zeta + C\left(B + \sqrt{(\Phi_{\overline{n}_\Lambda}(\mathcal{X}) + \log(1/\delta))}\right)\sqrt{\frac{\Phi_{\overline{n}_\Lambda}(\mathcal{X})}{\overline{n}_\Lambda}},$$

$$S_2 = C\left(B + \sqrt{\left(\Phi_{\underline{n}_\Lambda}(\hat{H}^\gamma) + \log(1/\delta)\right)}\right)\sqrt{\frac{\Phi_{\underline{n}_\Lambda}(\hat{H}^\gamma)}{\underline{n}_\Lambda}}.$$

Remark. 1. The regret rate of Corollary 4 is almost the same as Theorem 3, except the set $\mathcal{H}^\gamma$ is replaced with $\hat{H}^\gamma$. We note that all the terms in the regret rate depend only on $\bar{\zeta}$ or ζ, and do not depend on $\zeta_{\max}$. This means Algorithm 2 can adapt to unknown level of comparison bias ζ.
2. Similar to Theorem 3, Corollary 4 also requires the unknown quantity $\Phi_t(\hat{H}^\gamma)$ to set β_t; in practice we can also use a similar hyper-parameter search to find this quantity. γ also takes a similar effect as in Algorithm 1, and $\gamma = \frac{1}{L_2}\zeta_0$ can lead to $\hat{\mathcal{H}}^\gamma \approx \mathcal{H}$ and a practical algorithm. Again, setup of these parameters only depends on $\bar{\zeta}$ 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_c is a good approximation for f (in a weaker sense of being close in terms of f^* and f_c^*, 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_c: f_c is only available through comparisons, and we will get the same set of comparisons from f_c and $f_c + c$ 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_r 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 ζ known beforehand, whereas our modified COMP-GP-UCB (Algorithm 2) can adapt to an unknown ζ. 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 $(10^{-3}, 10^{1})$ and $(10^{-1}, 10^{5})$. 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 (1) w.r.t. f [2]. 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 $\frac{\lambda_l}{\lambda_c}$ the *cost ratio* between labels and comparisons; the larger the cost ratio, the cheaper the comparisons. Our algorithm generally works for a cost ratio $\frac{\lambda_l}{\lambda_c} > 1$. We test the performance under various cost ratios in our experiment. Unless otherwise specified, for all the experiments with a varying budget we follow the setup of MF-GP-UCB and use $\lambda_c = 0.1$ and $\lambda_l = 1$, and use a total budget of up to $\Lambda = 100$. 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 [3] 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).

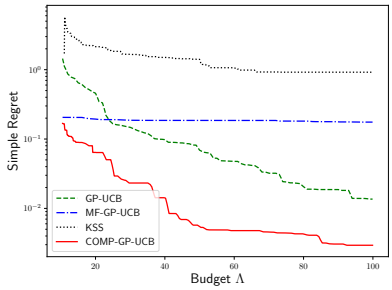
Then in Figure 1b, we fix the total budget to be $\Lambda = 100$ and cost of labels $\lambda_l = 1$, and vary the cost ratio from 1 to 10 by varying comparison costs. COMP-GP-UCB achieves the best performance for all setups except for $\lambda_c = \lambda_l = 1$ when it is worse than the label-only GP-UCB 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 (≥ 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

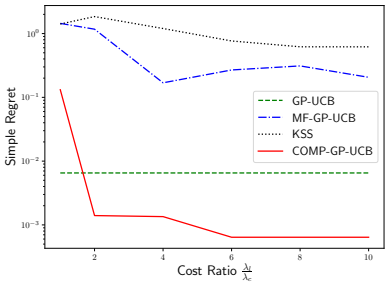
$$R_c(\Lambda) = \begin{cases} \min_{t, q_t = \text{comp}} f^* - f(x_t), & \text{if } \exists t \text{ s.t } q_t = \text{comp}, \\ \infty & \text{otherwise.} \end{cases}$$

[2] We find that KernelSelfSparring is extremely slow for $d = 8$ so we only test it for CurrinExp.

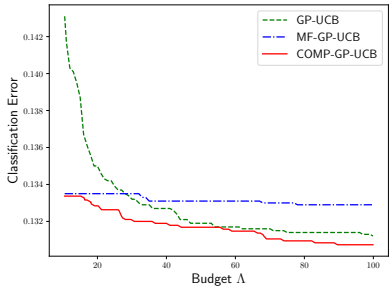
[3] Due to space limits, we include results on the Borehole function, along with other experiment results in the appendix.



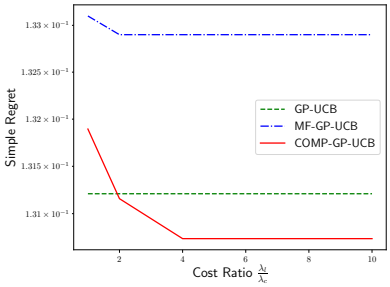
(a) CurrinExp, varying budget



(b) CurrinExp, varying cost ratio



(c) SVM Tuning, varying budget



(d) SVM Tuning, varying cost ratio

Figure 1: Empirical results comparing COMP-GP-UCB with baseline methods. KSS stands for KernelSelfSparring.

and

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

In Figure 2, we plot the comparison regret R_c and direct query regret R_l for the CurrinExp synthetic function (same setting as Figure 1a). We plot R_c and R_l respectively for both COMP-GP-UCB and MF-GP-UCB, R_l 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.

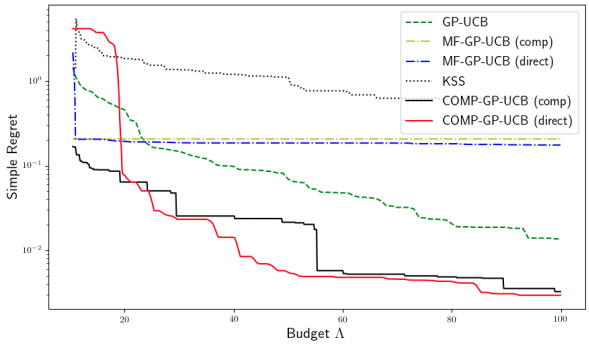


Figure 2: Result comparing comparison and direct regret.

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.

References

Ailon, N., Karnin, Z., and Joachims, T. (2014). Reducing dueling bandits to cardinal bandits. In *International Conference on Machine Learning*, pages 856–864.

Besbes, O. and Zeevi, A. (2009). Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57(6):1407–1420.

Bradley, R. A. and Terry, M. E. (1952). Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Bull, A. D. (2011). Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879–2904.

Chen, H. (1988). Lower rate of convergence for locating a maximum of a function. *The Annals of Statistics*, pages 1330–1334.

Chowdhury, S. R. and Gopalan, A. (2017). On kernelized multi-armed bandits. *arXiv preprint arXiv:1704.00445*.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Faber, F. A., Lindmaa, A., Von Lilienfeld, O. A., and Armiento, R. (2016). Machine learning energies of 2 million elpasolite (a b c 2 d 6) crystals. *Physical review letters*, 117(13):135502.

Flaxman, A. D., Kalai, A. T., and McMahan, H. B. (2005). Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 385–394. Society for Industrial and Applied Mathematics.

Ge, R., Huang, F., Jin, C., and Yuan, Y. (2015). Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842.

Hazan, E., Klivans, A., and Yuan, Y. (2017). Hyperparameter optimization: A spectral approach.

Heckel, R., Shah, N. B., Ramchandran, K., and Wainwright, M. J. (2016). Active ranking from pairwise comparisons and when parametric assumptions don't help. *arXiv preprint arXiv:1606.08842*.

Jamieson, K. G., Nowak, R., and Recht, B. (2012). Query complexity of derivative-free optimization. In *Advances in Neural Information Processing Systems*, pages 2672–2680.

Jones, D. R., Perttunen, C. D., and Stuckman, B. E. (1993). Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181.

Kandasamy, K., Dasarathy, G., Oliva, J. B., Schneider, J., and Poczos, B. (2016). Multi-fidelity gaussian process bandit optimisation. *arXiv preprint arXiv:1603.06288*.

Kandasamy, K., Dasarathy, G., Schneider, J., and Póczos, B. (2017). Multi-fidelity bayesian optimisation with continuous approximations. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 1799–1808. JMLR.org.

Kane, D. M., Lovett, S., Moran, S., and Zhang, J. (2017). Active classification with comparison queries. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 355–366. IEEE.

Kumagai, W. (2017). Regret analysis for continuous dueling bandit. In *Advances in Neural Information Processing Systems*, pages 1489–1498.

Malherbe, C., Contal, E., and Vayatis, N. (2016). A ranking approach to global optimization. In *International Conference on Machine Learning*, pages 1539–1547.

Radlinski, F., Kurup, M., and Joachims, T. (2008). How does clickthrough data reflect retrieval quality? In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 43–52. ACM.

Rusmevichientong, P. and Tsitsiklis, J. N. (2010). Linearly parameterized bandits. *Mathematics of Operations Research*, 35(2):395–411.

Sen, R., Kandasamy, K., and Shakkottai, S. (2018). Multi-fidelity black-box optimization with hierarchical partitions. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4538–4547, Stockholmsmässan, Stockholm Sweden. PMLR.

Smart, W. D. and Kaelbling, L. P. (2000). Practical reinforcement learning in continuous spaces. In *ICML*, pages 903–910.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. (2009). Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*.

Sui, Y., Zhuang, V., Burdick, J. W., and Yue, Y. (2017). Multi-dueling bandits with dependent arms. *arXiv preprint arXiv:1705.00253*.

Thurstone, L. L. (1927). A law of comparative judgment. *Psychological review*, 34(4):273.

Wang, Y., Balakrishnan, S., and Singh, A. (2018). Optimization of smooth functions with noisy observations: Local minimax rates. In *Advances in Neural Information Processing Systems*, pages 4338–4349.

Xiong, S., Qian, P. Z. G., and Wu, C. F. J. (2013). Sequential design and analysis of high-accuracy and low-accuracy computer codes. *Technometrics*, 55(1):37–46.

Xu, Y., Muthakana, H., Balakrishnan, S., Singh, A., and Dubrawski, A. (2018). Nonparametric regression with comparisons: Escaping the curse of dimensionality with ordinal information. In *International Conference on Machine Learning*, pages 5469–5478.

Xu, Y., Zhang, H., Miller, K., Singh, A., and Dubrawski, A. (2017). Noise-tolerant interactive learning from pairwise comparisons with near-minimal label complexity. *arXiv preprint arXiv:1704.05820*.

Xue, D., Balachandran, P. V., Hogden, J., Theiler, J., Xue, D., and Lookman, T. (2016). Accelerated search for materials with targeted properties by adaptive design. *Nature communications*, 7:11241.

Yue, Y., Broder, J., Kleinberg, R., and Joachims, T. (2012). The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556.

Zoghi, M., Karnin, Z. S., Whiteson, S., and de Rijke, M. (2015). Copeland dueling bandits. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 307–315. Curran Associates, Inc.