Efficient Batch Black-box Optimization with Deterministic Regret Bounds

Yueming Lyu*
University of Technology Sydney

Yuan Yuan
Hong Kong University of Science and Technology

Ivor Tsang
University of Technology Sydney

Abstract

In this work, we investigate black-box optimization from the perspective of frequentist kernel methods. We propose a novel batch optimization algorithm to jointly maximize the acquisition function and select points from a whole batch in a holistic way. Theoretically, we derive regret bounds for both the noise-free and perturbation settings. Moreover, we analyze the property of the adversarial regret that is required by robust initialization for Bayesian Optimization (BO), and prove that the adversarial regret bounds decrease with the decrease of covering radius, which provides a criterion for generating (initialization point set) to minimize the bound. We then propose fast searching algorithms to generate a point set with a small covering radius for the robust initialization. Experimental results on both synthetic benchmark problems and real-world problems show the effectiveness of the proposed algorithms.

1 Introduction

Bayesian Optimization (BO) is a promising approach to address the expensive black-box (non-convex) optimization problems. Applications of BO include: automatic tuning of hyper-parameters in machine learning [26], the gait optimization in robot control [21], identifying molecular compounds in drug discovery [23], and optimizing of computation-intensive engineering design [28].

BO aims to find the optimum of an unknown, usually non-convex function $f$. Since little information is known about the underlying function $f$, BO requires to estimate a surrogate function to model the unknown function. Therefore, one major challenge of BO, is to seek a balance between collecting information to model the function $f$ (exploration) and searching for an optimum based on collected information (exploitation).

Typically, BO assumes that the underlying function $f$ is sampled from a Gaussian process (GP) prior. BO selects the candidate solutions for evaluation by maximizing an acquisition function [20, 22, 15]), which balances the exploration and exploitation, given all previous observations. In practice, BO can usually find an approximate maximum solution with a remarkably small number of function evaluations [26, 24].

In many real applications, it is usually preferred that multiple function evaluations are performed in parallel to achieve time efficiency, for example, examining various hyperparameter settings of a machine learning algorithm simultaneously or running multiple instances of a reinforcement learning simulation in parallel. Sequential BO selection is by no means efficient in these scenarios. Therefore, several batch BO approaches have been proposed to address this issue. Shah et al. [25] propose a parallel predictive entropy search method, which is based on the predictive entropy search (PES) [13] and adapts the method to the batch case. Wu et al. [31] generalize the knowledge gradient method [10] to a parallel knowledge gradient method. These methods are, however, computationally intensive due

*LV_Yueming@outlook.com

Preprint. Working in Progress.
to relying on Monte Carlo sampling. In addition, they are not scalable to a large batch size and they lack the theoretical convergence guarantee.

Instead of using Monte Carlo sampling, another line of research improves the efficiency by deriving a tighter upper confidence bound. The GP-BUCB policy \[9\] makes selections point by point sequentially, according to an upper confidence bound (UCB) criterion \([2, 27]\) with a fixed mean function and an updated covariance function, until reaching the preset batch size. It proves sublinear growth bounds on the cumulative regret, which guarantee the bound on the number of required iterations to get close enough to the optimal. The GP-UCB-PE \([7]\) combines the UCB strategy and the pure exploration \([5]\) in the evaluations of the same batch. The GP-UCB-PE achieves a better upper bound on the cumulative regret compared with the GP-BUCB. However, although these methods do not require Monte Carlo sampling, they select candidate queries of a batch in a greedy manner. As a result, they are still far from satisfactory in terms of efficiency and scalability.

To achieve a greater efficiency in batch selection, we propose to simultaneously select candidate queries of a batch in a holistic manner, rather than the previous sequential manner. In this paper, we analyze both batch BO and sequential BO from a frequentist perspective. For the batch BO, we propose a novel batch selection method which takes both the mean prediction value and correlation of points in a batch into consideration. By jointly maximizing the novel acquisition function with respect to all the points in a batch, the proposed method is able to attain a better exploitation/exploration trade off. For the sequential BO, we obtain a similar acquisition function as that in the GP-UCB \([27]\), except that our function employs a constant weight for the deviation term. The constant weight is preferred over the previous theoretical weight proposed in GP-UCB, because the latter is overly conservative, which has been observed in many other works \([4, 3, 27]\). Moreover, for functions with a bounded norm in the reproducing kernel Hilbert space (RKHS), we derive the non-trivial regret bounds for both the batch BO method and the sequential BO method.

At the beginning of the BO, as little information is known, the initialization phase becomes vitally important. To obtain a good and robust initialization, we first study the properties that need to be satisfied for a robust initialization through analyzing the adversarial regret. We prove that the regret bounds decrease with the decrease of the covering radius (named fill distance in \([16]\)). As minimizing the covering radius of a lattice is equivalent to maximizing its packing radius (named separate distance in \([16]\)) \([8, 17]\), we then propose a novel fast searching method to maximize the packing radius of a rank-1 lattice and obtain the points set with a small covering radius.

**Limitations of Bull’s batch method:** Bull \([6]\) presents a non-adaptive batch method with all the query points except one being fixed at the beginning. However, as mentioned by Bull, this method is not practical. We propose an adaptive BO method and initialize it with a robust initialization algorithm. More specifically, we first select the initialization query points by minimizing the covering radius, and then select the query points based on our adaptive methods.

**Relationship to Bull’s bounds:** We give the regret bound w.r.t. the covering radius for different kernels; while Bull’s bound is limited to Marténn type kernel. Compared with Bull’s bound (Theorem 1 in \([6]\)), our regret bound directly links to the covering radius (fill distance), which provides a criterion for generating a point set to achieve small bounds. In contrast, Bull’s bound does not provide a criterion for minimizing the bound. We generate the initialization point set by minimizing covering radius (our bound); while Bull’s work doesn’t.

Our contributions are summarized as follows:

- We study black-box optimization for functions with a bounded norm in RKHS and achieve deterministic regret bounds for both the noise-free setting and the perturbation setting. The study not only brings more insight into the BO literature, but also provides a better guidance for designing new acquisition functions.

- We propose more-efficient novel adaptive algorithms for batch optimization, which select candidate queries of a batch in a holistic manner. Theoretically, we prove that the proposed methods achieve non-trivial regret bounds.

- We analyze the adversarial regret for a robust initialization of BO, and theoretically prove that the regret bounds decrease with the decrease of the covering radius, which provides a criterion for generating points set to minimize the bound (for initialization of BO).
• We propose novel, fast searching algorithms to maximize the packing radius of a rank-1 lattice and generate a set of points with a small covering radius. The generated points set provides a robust initialization for BO. Moreover, the set of points can be used for integral approximation on domain $[0, 1]^d$. Experimental results show that the proposed method can achieve a larger packing radius (separate distance) compared with baselines.

2 Notations and Symbols

Let $\mathcal{H}_k$ denote a separable reproducing kernel Hilbert space associated with the kernel $k(\cdot, \cdot)$, and Let $\| \cdot \|_{\mathcal{H}_k}$ denote the RKHS norm in $\mathcal{H}_k$. \| \cdot \| denotes the $l_2$ norm (Euclidean distance). Let $B_k = \{ f : f \in \mathcal{H}_k, \| f \|_{\mathcal{H}_k} \leq B \}$ denotes a bounded subset in the RKHS, and $\mathcal{X} \subset \mathbb{R}^d$ denote a compact set in $\mathbb{R}^d$. Symbol $[N]$ denotes the set $\{1, \ldots, N\}$. $\mathbb{N}$ and $\mathbb{P}$ denote the integer set and prime number set, respectively. Bold capital letters are used for matrices.

3 Background and Problem Setup

Let $f : \mathcal{X} \to \mathbb{R}$ be the unknown black-box function to be optimized, where $\mathcal{X} \subset \mathbb{R}^d$ is a compact set. BO aims to find a maximum $x^*$ of the function $f$, i.e., $f(x^*) = \max_{x \in \mathcal{X}} f(x)$.

In sequential BO, a single point $x_t \in \mathcal{X}$ is selected to query an observation at round $t$. Batch BO is introduced in the literature for the benefits of parallel execution. Batch BO methods select a batch of points $X_n = \{ x_{(n-1)L+1}, \ldots, x_{nL} \}$ simultaneously at round $n$, where $L$ is the batch size. The batch BO is different from the sequential BO because the observation is delayed for batch BO during the batch selection phase. An additional challenge is introduced in batch BO since it needs to select a batch of points at one time, without knowing the latest information about the function $f$.

In BO, the effectiveness of a selection policy can be measured by the cumulative regret $R_T$ and the simple regret (minimum regret) $r_T$ over $T$ steps. The cumulative regret $R_T$ and simple regret $r_T$ are defined in equations (1) and (2), respectively:

$$r_T = f(x^*) - \max_{1 \leq t \leq T} f(x_t)$$

$$R_T = \sum_{t=1}^{T} (f(x^*) - f(x_t))$$

The regret bound introduced in numerous theoretical works is based on the maximum information gain defined as

$$\gamma_T = \max_{x_1, \ldots, x_T} \frac{1}{2} \log \det (I_T + \sigma^{-2} K_T).$$

The bounds of $\gamma_T$ for commonly used kernels are studied in [27]. Specifically, Srinivas et al. [27] state that $\gamma_T = O(d \log T)$ for the linear kernel, $\gamma_T = O((\log T)^{d+1})$ for the squared exponential kernel and $\gamma_T = O(T^\alpha (\log T))$ for the Matérn kernels with $\nu > 1$, where $\alpha = \frac{d(d+1)}{2(d+1)} \leq 1$. We employ the term $\gamma_T$ to build the regret bounds of our algorithms.

In this work, we consider two settings: the noise-free setting and perturbation setting.

Noise-Free Setting: We assume the underlying function $f$ belongs to an RKHS associated with kernel $k(\cdot, \cdot)$, i.e., $f \in \mathcal{H}_k$, with $\| f \|_{\mathcal{H}_k} < \infty$. In the noise-free setting, we can directly observe $f(x), x \in \mathcal{X}$ without noise perturbation.

Perturbation Setting: In the perturbation setting, we can not observe the function evaluation $f(x)$ directly. Instead, we observe $y = h(x) = f(x) + g(x)$, where $g(x)$ is an unknown perturbation function.

Define $k^\sigma(x, y) := k(x, y) + \sigma^2 \delta(x, y)$ for $x, y \in \mathcal{X}$, where $\delta(x, y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$ and $\sigma \geq 0$. We assume $f \in \mathcal{H}_k, g \in \mathcal{H}_{\sigma^2}$ with $\| f \|_{\mathcal{H}_k} < \infty$ and $\| g \|_{\mathcal{H}_{\sigma^2}} < \infty$, respectively. Therefore, we know $h \in \mathcal{H}_{\sigma^2}$ and $\| h \|_{\mathcal{H}_{\sigma^2}} < \infty$. 

3
Algorithm 1

for \( t = 1 \) to \( T \) do
  Obtain \( m_{t-1}(\cdot) \) and \( \sigma^2_{t-1}(\cdot) \) via equations (4) and (5).
  Choose \( x_t = \arg\max_{x \in X} m_{t-1}(x) + \|f\|_{\mathcal{H}_k} \sigma_{t-1}(x) \).
  Query the observation \( f(x_t) \) at location \( x_t \).
end for

4 BO in Noise-Free Setting

In this section, we will first present algorithms and theoretical analysis in the sequential case. We then discuss our batch selection method. All detailed proofs are included in the supplementary material.

4.1 Sequential Selection in Noise Free Setting

Define \( m_t(x) \) and \( \sigma_t(x) \) as follows:

\[
m_t(x) = k_t(x)^T K_t^{-1} f_t
\]

\[
\sigma^2_t(x) = k_t(x, x) - k_t(x)^T K_t^{-1} k_t(x),
\]

(4)

where \( k_t(x) = [k(x, x_1), \ldots, k(x, x_t)]^T \) and the kernel matrix \( K_t = [k(x_i, x_j)]_{1 \leq i, j \leq t} \). These terms are closely related to the posterior mean and variance functions of GP with zero noise. We use them in the deterministic setting. A detailed review of the relationships between GP methods and kernel methods can be found in [16].

The sequential optimization method in the noise-free setting is described in Algorithm 1. It has a similar form to GP-UCB [27], except that it employs a constant weight of the term \( \sigma_1 \) to balance exploration and exploitation. In practice, a constant weight is preferred in the scenarios where an aggressive selection manner is needed. For example, only a small number of evaluations can be done in the hyperparameter tuning practice, a constant weight is preferred in the scenarios where an aggressive selection manner is needed. For example, only a small number of evaluations can be done in the hyperparameter tuning practice.

**Theorem 1.** Suppose \( f \in \mathcal{H}_k \) associated with \( k(x, \cdot) \leq 1 \) and \( ||f||_{\mathcal{H}_k} < \infty \). Let \( C_1 = \frac{s}{\log(1 + \sigma - \gamma)} \).

Algorithm 1 achieves a cumulative regret bound and a simple regret bound given as follows:

\[
R_T \leq \|f\|_{\mathcal{H}_k} \sqrt{T C_1 \gamma T},
\]

(6)

\[
r_T \leq \|f\|_{\mathcal{H}_k} \sqrt{C_1 \gamma T}. \tag{7}
\]

**Remark:** We can achieve concrete bounds w.r.t \( T \) by replacing \( \gamma T \) with the specific bound for the corresponding kernel. For example, for SE kernels, we can obtain that \( R_T = O(\sqrt{T}(\log T)^{d+1}) \) and \( r_T = O(\frac{\log T}{\sqrt{T}}) \), respectively. Bull [6] presents bounds for Matérn type kernels. The bound in Theorem 1 is tighter than Bull’s bound of pure EI (Theorem 4 in [6]) when the smoothness parameter of the Matérn kernel \( \nu > \frac{d(d+1)}{d-2} = O(d) \). This is no better than the bound of mixed strategies (Theorem 5) in Bull’s work. Nevertheless, the bound in Theorem 1 makes fewer assumptions about the kernels, and covers more general results (kernels) compared with Bull’s work.

4.2 Batch Selection in Noise-Free Setting

Let \( N \) and \( L \) be the number of batches and the batch size, respectively. Without loss of generality, we assume \( T = NL \). Let \( X_n = \{x_{(n-1)L+1}, \ldots, x_{nL}\} \) and \( \overline{X}_n = \{X_1, \ldots, X_n\} = \{x_1, \ldots, x_{nL}\} \) be the \( n \)th batch of points and all the \( n \) batches of points, respectively. The covariance function of \( X \in \mathbb{R}^{d \times L} \) for the noise free case is given in equation (8):

\[
cov_n(X, X) = K(X, X) - K(\overline{X}_n, X)^T K(\overline{X}_n, \overline{X}_n)^{-1} K(\overline{X}_n, X)
\]

(8)

where \( K(X, X) \) is the \( L \times L \) kernel matrix, \( K(\overline{X}_n, X) \) denotes the \( nL \times L \) kernel matrix between \( \overline{X}_n \) and \( X \). When \( n = 0 \), \( \text{cov}_0(X, X) = K(X, X) \) is the prior kernel matrix. We assume that the kernel matrix is invertible in the noise-free setting.
Algorithm 2

\begin{algorithm}
\begin{algorithmic}
\FOR{$n = 1$ \TO $N$}
\STATE Obtain $m_{(n-1)L}()$ and $\text{cov}_{n-1}()$ via equations (4) and (8), respectively.
\STATE Choose $X_n = \arg \max_{X \subset \mathcal{X}} \frac{1}{L} \sum_{t=1}^{L} m_{(n-1)L}(X, i) + \| f \|_{\mathcal{H}_k} \left( 2 \sqrt{\frac{\text{tr}(\text{cov}_{n-1}(X,X))}{L}} - \sqrt{\frac{\text{cov}_{n-1}(X,X)1}{L}^T} \right)$.
\STATE Query the batch observations $\{ f(x_{(n-1)L+1}), ..., f(x_{nL}) \}$ at locations $X_n$.
\ENDFOR
\end{algorithmic}
\end{algorithm}

The proposed batch optimization algorithm is presented in Algorithm 2. It employs the mean prediction value of a batch together with a term of covariance to balance the exploration/exploitation trade-off. The covariance term in Algorithm 2 penalizes the batch with over-correlated points. Intuitively, for SE kernels and Matérn kernels, it penalizes the batch with points that are too close to each other (w.r.t Euclidean distance). As a result, it encourages the points in a batch to spread out for a better exploration. The regret bounds of our batch optimization method are summarized in Theorem 2.

**Theorem 2.** Suppose $f \in \mathcal{H}_k$ associated with $k(x, x) \leq 1$ and $\| f \|_{\mathcal{H}_k} < \infty$. Let $T = NL$, $\beta = \max_{n \in [1, ..., N]} \| \text{cov}_{n-1}(X_n, X_n) \|_2$ and $C_2 = \frac{8\beta}{\log(1 + \beta \sigma^{-2})}$. Algorithm 2 with batch size $L$ achieves a cumulative regret bound and a simple regret bound given by equations (9) and (10), respectively:

\begin{align*}
R_T &\leq \| f \|_{\mathcal{H}_k} \sqrt{TC_2} \gamma T \quad (9) \\
&\leq \| f \|_{\mathcal{H}_k} \sqrt{C_2} \gamma \sqrt{T}. \quad (10)
\end{align*}

**Remark:** (1) A large $\beta$ leads to a large bound, while a small $\beta$ attains a small bound. Algorithm 2 punishes the correlated points and encourages the uncorrelated points in a batch, which can attain a small $\beta$ in general. (2) A trivial bound of $\beta$ is $\beta \leq L$.

To prove Theorem 2, the following Lemma is proposed. The detailed proof can be found in the supplementary material.

**Lemma 1.** Suppose $f \in \mathcal{H}_k$ associated with kernel $k(x, x)$ and $\| f \|_{\mathcal{H}_k} < \infty$, then

\begin{align*}
\left( \sum_{i=1}^{L} m_t(\hat{x}_i) - \sum_{i=1}^{L} f(\hat{x}_i) \right)^2 &\leq \| f \|_{\mathcal{H}_k}^2 (1^T A 1), \quad \text{where } A \text{ denotes the kernel covariance matrix with } A_{ij} = k(\hat{x}_i, \hat{x}_j) - k_t(\hat{x}_i) k_t(\hat{x}_j). \\
\end{align*}

**Remark:** Lemma 1 provides a tighter bound for the deviation of the summation of a batch than directly applying the bound for a single point $L$ times.

\section{BO in Perturbation Setting}

In the perturbation setting, we cannot observe the function evaluation $f(x)$ directly. Instead, we observe $y = h(x) = f(x) + g(x)$, where $g(x)$ is an unknown perturbation function. We will discuss the sequential selection and batch selection methods in the following sections, respectively.

\subsection{Sequential Selection in Perturbation Setting}

Define $\hat{m}_t(x)$ and $\hat{\sigma}_t(x)$ as follows:

\begin{align*}
\hat{m}_t(x) &= k_t(x)^T (K_t + \sigma^2 I)^{-1} y_t \quad (11) \\
\hat{\sigma}_t^2(x) &= k(x, x) - k_t(x)^T (K_t + \sigma^2 I)^{-1} k_t(x), \quad (12)
\end{align*}

where $k_t(x) = [k(x, x_1), ..., k(x, x_L)]^T$ and the kernel matrix $K_t = [k(x_i, x_j)]_{1 \leq i, j \leq t}$.

The sequential selection method is presented in Algorithm 3. It has a similar formula to Algorithm 1 while Algorithm 3 employs an regularization $\sigma^2 I$ to handle the uncertainty of the perturbation. The regret bounds of Algorithm 3 are summarized in Theorem 5.
Algorithm 3

for $t = 1$ to $T$

Obtain $\hat{m}_{t-1}(\cdot)$ and $\hat{\sigma}_t^2(\cdot)$ via equation (11) and (12).
Choose $x_t = \arg\max_{x \in X} \hat{m}_{t-1}(x) + ||h||_{\mathcal{H}_k} \hat{\sigma}_{t-1}(x)$

Query the observation $y_t = h(x_t)$ at location $x_t$.

end for

Theorem 3. Define $k^\sigma(x, y) := k(x, y) + \sigma^2 \delta(x, y) \leq B$, where $\delta(x, y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$ and $\sigma \geq 0$. Suppose $f \in \mathcal{H}_k$, $g \in \mathcal{H}_{\sigma^2 \delta}$ associated with kernel $k$ and kernel $\sigma^2 \delta$ with $\|f\|_{\mathcal{H}_k} < \infty$ and $\|g\|_{\mathcal{H}_{\sigma^2 \delta}} < \infty$, respectively. Let $C_3 = \frac{8B}{\log(1 + B \sigma^2 \delta)}$. Algorithm 3 achieves a cumulative regret bound and a simple regret bound given by equations (13) and (14), respectively.

$$R_T \leq \|h\|_{\mathcal{H}_k} \sqrt{T C_3 T} + 2T \left(\|h\|_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_{\sigma^2 \delta}}\right) \sigma$$

(13)

$$r_T \leq \|h\|_{\mathcal{H}_k} \sqrt{\frac{C_3 \gamma T}{T}} + 2 \left(\|h\|_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_{\sigma^2 \delta}}\right) \sigma$$

(14)

Remark: In the perturbation setting, the unknown perturbation function $g$ results in an unavoidable regret term with respect to $\sigma$ in the regret bound compared with GP-UCB [27]. Note that the bounds in [27] are probabilistic bounds. There is always a positive probability that the bounds in [27] fail. In contrast, the bounds in Theorem 3 are deterministic.

Corollary 1. Suppose $h = f \in \mathcal{H}_k$ associated with $k(x, y) \leq 1$ and $\|f\|_{\mathcal{H}_k} < \infty$. Let $C_1 = \frac{8}{\log(1 + \sigma^2 \delta)}$. Algorithm 3 achieves a cumulative regret bound and a simple regret bound given by equations (15) and (16), respectively.

$$R_T \leq \|f\|_{\mathcal{H}_k} \sqrt{T C_1 \gamma T} + 2T \|f\|_{\mathcal{H}_k} \sigma$$

(15)

$$r_T \leq \|f\|_{\mathcal{H}_k} \sqrt{\frac{C_1 \gamma T}{T}} + 2 \|f\|_{\mathcal{H}_k} \sigma.$$  

(16)

Proof. Setting $g = 0$ and $B = 1$ in Theorem 3 we can achieve the results.

Remark: In practice, a small constant $\sigma^2 I$ is added to the kernel matrix to avoid numeric problems in the noise-free setting. Corollary 1 shows that the small constant results in an additional biased term in the regret bound. Theorem 1 employs (11) and (12) for updating, while Corollary 1 presents the regret bound for the practical updating by (13) and (14).

5.2 Batch Selection in Perturbation Setting

The covariance kernel function of $X \in \mathbb{R}^{d \times L}$ for the perturbation setting is defined as equation (17).

$$\text{cov}_n(X, Y) = K(X, Y) - K(X_n, X)^T (\sigma^2 I + K(X_n, X_n)^{-1}) K(X_n, X), \quad (17)$$

where $K(X, Y)$ is the $L \times L$ kernel matrix, and $K(X_n, X)$ denotes the $nL \times L$ kernel matrix between $X_n$ and $X$. The batch optimization method for the perturbation setting is presented in Algorithm 4. The regret bounds of Algorithm 4 are summarized in Theorem 4.

Theorem 4. Define $k^\sigma(x, y) := k(x, y) + \sigma^2 \delta(x, y) \leq B$, where $\delta(x, y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$ and $\sigma \geq 0$. Suppose $f \in \mathcal{H}_k$ and $g \in \mathcal{H}_{\sigma^2 \delta}$ associated with kernel $k$ and kernel $\sigma^2 \delta$ with $\|f\|_{\mathcal{H}_k} < \infty$ and $\|g\|_{\mathcal{H}_{\sigma^2 \delta}} < \infty$, respectively. Let $T = NL$, $\beta = \max_{n \in \{1, \ldots, N\}} \|\text{cov}_{n-1}(X_n, X_n)\|_2$ and $C_4 = \frac{\beta}{\log(1 + \beta \sigma^2 \delta)}$. Algorithm 4 with batch size $L$ achieves a cumulative regret bound and a simple regret bound given by equations (18) and (19), respectively.

$$R_T \leq \|h\|_{\mathcal{H}_k} \sqrt{T C_4 \gamma T} + 2T \left(\|h\|_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_{\sigma^2 \delta}}\right) \sigma$$

(18)

$$r_T \leq \|h\|_{\mathcal{H}_k} \sqrt{\frac{C_4 \gamma T}{T}} + 2 \left(\|h\|_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_{\sigma^2 \delta}}\right) \sigma.$$  

(19)

Remark: When the batch size is one, the regret bounds reduce to the sequential case.
Algorithm 4

\begin{algorithm}
\caption{Approximating \(\hat{m}\) and \(\hat{\text{cov}}\) in the acquisition function.}
\begin{algorithmic}
\FOR{n = 1 \TO N}
\STATE Obtain \(\hat{m}_{n-1}(\cdot)\) and \(\hat{\text{cov}}_{n-1}(\cdot)\) via equation (11) and (17) respectively.
\STATE \textbf{Choose} \(X_n = \arg\max_{X \subseteq \mathcal{X}} \sum_{i=1}^L \hat{m}_{n-1}(X_{\cdot,i}) + \|h\|_{\mathcal{H}_k} \sigma \left(2 \sqrt{\frac{\text{tr}(\hat{\text{cov}}_{n-1}(X,X))}{L}} - \sqrt{\frac{1}{L^2} \text{tr}(\hat{\text{cov}}_{n-1}(X,X))}\right)\)
\STATE Query the batch observations \(\{h(x_{(n-1)}L+1), \ldots, h(x_{nL})\}\) at locations \(X_n = \{x_{(n-1)}L+1, \ldots, x_{nL}\}\).
\ENDFOR
\end{algorithmic}
\end{algorithm}

6 Experiments

In this section, we focus on the evaluation of the proposed batch method. We evaluate the proposed Batch kernel optimization (BKOP) by comparing it with GP-BUCB [9] and GP-UCB-PE[7] on several synthetic benchmark test problems, hyperparameters tuning of a deep network on CIFAR100 [19] and the robot pushing task in [29]. An empirical study of our fast rank-1 lattice searching method is included in the supplementary material.

Synthetic benchmark problems: The synthetic test functions and the domains employed are listed in Table 4 in the supplementary material, which includes nonconvex, nonsmooth and multimodal functions.

We fix the weight of the covariance term in the acquisition function of BKOP to one in all the experiments. For all the synthetic test problems, we set the dimension of the domain \(d = 6\), and we set the batch size to \(L = 5\) and \(L = 10\) for all the batch BO algorithms. We use the the ARD Matérn 5/2 kernel for all the methods. Instead of finding the optimum by discrete approximation, we employ the CMA-ES algorithm [12] to optimize the acquisition function in the continuous domain \(\mathcal{X}\) for all the methods, which usually improves the performance compared with discrete approximation. For each test problem, we use 20 rank-1 lattice points resized in the domain \(\mathcal{X}\) as the initialization. All the methods use the same initial points.

The mean value and error bar of the simple regret over 30 independent runs with respect to different algorithms are presented in Figure 1. We can observe that BKOP with batch size 5 and 10 performs better than the other methods with the same batch size. Moreover, algorithms with batch size 5
achieve faster decreasing regret compared with batch size 10. BKOP achieves significantly low regret compared with the other methods on the Different-Powers and Rosenbrock test functions.

**Hyperparameter tuning of network:** We evaluate BKOP on hyperparameter tuning of the network on the CIFAR100 dataset. The network we employ contains three hidden building blocks, each one consisting of one convolution layer, one batch normalization layer and one Relu layer. The depth of a building block is defined as the repeat number of these three layers. Seven hyperparameters are used in total for searching, namely, the depth of the building block ($\{1, 2, 3\}$), the initialized learning rate for SGD ($[10^{-4}, 10^{-1}]$), the momentum weight ($[0.1, 0.95]$), weight of L2 regularization ($[10^{-10}, 10^{-2}]$), and three hyperparameters related to the filter size for each building block, the domain of these three parameters is $\{2 \times 2, 3 \times 3, 4 \times 4\}$. We employ the default training set (i.e., 50,000 samples) for training, and use the default test set (i.e., 10,000 samples) to compute the validation error regret of automatic hyperparameter tuning for all the methods.

We employ five rank-1 lattice points resized in the domain as the initialization. All the methods use the same initial points. The mean value of the simple regret of the validation error in percentage terms over 10 independent runs is presented in Figure 2(a). We can observe that BKOP with both batch size 5 and 10 outperforms the others. Moreover, the performance of GP-UCB-PE with batch size 10 is worse than the others.

**Robot Pushing Task:** We further evaluate the performance of BKOP on the robot pushing task in [29]. The goal of this task is to select a good action for pushing an object to a target location. The 4-dimensional robot pushing problem consists of the robot location $(x, y)$ and angle $\theta$ and the pushing duration $\tau$ as the input. And it outputs the distance between the pushed object and the target location as the function value. We employ 20 rank-1 lattice points as initialization. All the methods use the same initialization points. Thirty goal locations are randomly generated for testing. All the methods use the same goal locations. The mean value and error bars over 30 trials are presented in Figure 2(b). We can observe that BKOP with both batch size 5 and batch size 10 can achieve lower regret compared with GP-BUCB and GP-UCB-PE.

7 Conclusion

We analyzed black box optimization for functions with a bounded norm in RKHS. For sequential BO, we obtain a similar acquisition function to GP-UCB, but with a constant deviation weight. For batch BO, we proposed the BKOP algorithm, which is competitive with, or better than, other batch confidence-bound methods on a variety of tasks. Theoretically, we derive regret bounds for both the sequential case and batch case. Furthermore, we derive adversarial regret bounds with respect to the covering radius. We proposed fast searching methods to construct a good rank-1 lattice. Empirically, the proposed searching methods can obtain a large packing radius (separate distance).
References

[1] Dirk Nuyens Adrian Ebert, Hernan Leövey. Successive coordinate search and component-by-
component construction of rank-1 lattice rules. arXiv preprint arXiv:1703.06334, 2018.

[2] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. Journal of Machine
Learning Research, 3(Nov):397–422, 2002.

[3] Ilija Bogunovic, Jonathan Scarlett, Stefanie Jegelka, and Volkan Cevher. Adversarially robust
optimization with gaussian processes. In Advances in Neural Information Processing Systems,
pages 5765–5775, 2018.

[4] Ilija Bogunovic, Jonathan Scarlett, Andreas Krause, and Volkan Cevher. Truncated variance
reduction: A unified approach to bayesian optimization and level-set estimation. In Advances in
Neural Information Processing Systems, pages 1507–1515, 2016.

[5] Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits
problems. In International conference on Algorithmic learning theory, pages 23–37. Springer,
2009.

[6] Adam D Bull. Convergence rates of efficient global optimization algorithms. Journal of
Machine Learning Research, 12(Oct):2879–2904, 2011.

[7] Emile Contal, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis. Parallel gaussian
process optimization with upper confidence bound and pure exploration. In Joint European
Conference on Machine Learning and Knowledge Discovery in Databases, pages 225–240.
Springer, 2013.

[8] Sabrina Dammertz and Alexander Keller. Image synthesis by rank-1 lattices. In Monte Carlo
and Quasi-Monte Carlo Methods 2006, pages 217–236. Springer, 2008.

[9] Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation
tradeoffs in gaussian process bandit optimization. Journal of Machine Learning Research,
15(1):3873–3923, 2014.

[10] Peter Frazier, Warren Powell, and Savas Dayanik. The knowledge-gradient policy for correlated
normal beliefs. INFORMS journal on Computing, 21(4):599–613, 2009.

[11] Leonhard Grünschloß, Johannes Hanika, Ronnie Schwede, and Alexander Keller. (t, m, s)-nets
and maximized minimum distance. In Monte Carlo and Quasi-Monte Carlo Methods 2006,
pages 397–412. Springer, 2008.

[12] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of
the derandomized evolution strategy with covariance matrix adaptation (cma-es). Evolutionary
computation, 11(1):1–18, 2003.

[13] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive
entropy search for efficient global optimization of black-box functions. In NIPS, pages 918–926,
2014.

[14] L-K Hua and Yuan Wang. Applications of number theory to numerical analysis. Springer
Science & Business Media, 2012.

[15] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of
expensive black-box functions. Journal of Global optimization, 13(4):455–492, 1998.

[16] Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K Sriperumbudur. Gaussian
processes and kernel methods: A review on connections and equivalences. arXiv preprint
arXiv:1807.02582, 2018.

[17] Alexander Keller, Stefan Heinrich, and Harald Niederreiter. Monte Carlo and Quasi-Monte
Carlo Methods 2006. Springer, 2007.

[18] N. M. Korobov. Properties and calculation of optimal coefficients. Dokl. Akad. Nauk SSSR,
132:1009–1012, 1960.
[19] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[20] Harold J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86(1):97–106, 1964.

[21] Daniel J Lizotte, Tao Wang, Michael H Bowling, and Dale Schuurmans. Automatic gait optimization with gaussian process regression. In *IJCAI*, volume 7, pages 944–949, 2007.

[22] Jonas Močkus. On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404. Springer, 1975.

[23] Diana M Negoescu, Peter I Frazier, and Warren B Powell. The knowledge-gradient algorithm for sequencing experiments in drug discovery. *INFORMS Journal on Computing*, 23(3):346–363, 2011.

[24] Jonathan Scarlett. Tight regret bounds for bayesian optimization in one dimension. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4500–4508, 2018.

[25] Amar Shah and Zoubin Ghahramani. Parallel predictive entropy search for batch global optimization of expensive objective functions. In *NIPS*, pages 3330–3338, 2015.

[26] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *NIPS*, pages 2951–2959, 2012.

[27] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. 2010.

[28] G Gary Wang and Songqing Shan. Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical design*, 129(4):370–380, 2007.

[29] Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient bayesian optimization. In *International Conference on Machine Learning (ICML)*, page 3627–3635, 2017.

[30] Holger Wendland. *Scattered data approximation*, volume 17. Cambridge university press, 2004.

[31] Jian Wu and Peter Frazier. The parallel knowledge gradient method for batch bayesian optimization. In *NIPS*, pages 3126–3134, 2016.
A Robust Initialization for BO

In this section, we will discuss how to achieve robust initialization by analyzing regret in the adversarial setting. We will show that algorithms that attain a small covering radius (fill distance) are able to achieve small adversarial regret bounds.

Let \( f_t : \mathcal{X} \to \mathbb{R} \), \( t \in [T] \) be the black-box function to be optimized at round \( t \). Let \( f_t(x_i^*) = \max_{x \in \mathcal{X}} f_t(x) \) with \( f_t \in B_k \). The simple adversarial regret \( \bar{r}_T \) is defined as:

\[
\bar{r}_T = \min_{t \in [T]} \sup_{f_t \in B_k, \forall i \in [t-1]} \left\{ f_t(x_i^*) - f_t(x_i) \right\},
\]

where the constraints ensure that each \( f_t \) has the same observation values as the history at previous query points \( X_{t-1} = \{x_1, \ldots, x_{t-1}\} \). This can be viewed as an adversarial game. During each round \( t \), the opponent chooses a function \( f_t \) from a candidate set, and we then choose a query \( x_i \) in order to achieve a small regret. A robust initialization setting can be viewed as the batch of points that can achieve low simple adversarial regret irrespective of the access order.

Define covering radius (fill distance) and packing radius (separate distance) of a set of points \( X = \{x_1, \ldots, x_T\} \) as equations (21) and (22), respectively:

\[
h_X = \sup_{x \in \mathcal{X}} \min_{x_i \in \mathcal{X}} \|x - x_i\|
\]

\[
\rho_X = \frac{1}{2} \min_{x_i, x_j \in \mathcal{X}} \|x_i - x_j\|.
\]

Our method for robust initialization is presented in Algorithm 5, which constructs an initialization set \( X_{T-1} \) by minimizing the covering radius. We present one such method in Algorithm 6 in the next section. The initialization set \( X_{T-1} \) can be evaluated in a batch manner, which is able to benefit from parallel evaluation. The regret bounds of Algorithm 5 are summarized in Theorem 5 and Theorem 6.

**Theorem 5.** Define \( B_k = \{ f : f \in \mathcal{H}_k, \|f\|_{\mathcal{H}_k} \leq B \} \) associated with \( k(x, x) \) for \( x \in \mathcal{X} \subset \mathbb{R}^d \). Suppose \( f \in B_k \) and \( \mathcal{H}_k \) is norm-equivalent to the Sobolev space of order \( s \). Then there exists a constant \( C > 0 \), such that the query point set generated by Algorithm 5 with a sufficiently small covering radius (fill distance) \( h_X \) achieves a regret bound given by equation (23):

\[
\bar{r}_T \leq B Ch_X^{s-d/2}.
\]

**Remark:** The regret bound decreases as the covering radius becomes smaller. This means that a query set with a small covering radius can guarantee a small regret. Bull [6] gives bounds of fixed points set for Matérn kernels (Theorem 1). However, it does not link to the covering radius. The bound in Theorem 5 directly links to the covering radius, which provides a criterion for generating points to achieve small bounds.

**Theorem 6.** Define \( B_k = \{ f : f \in \mathcal{H}_k, \|f\|_{\mathcal{H}_k} \leq B \} \) associated with square-exponential \( k(x, x) \) on unit cube \( \mathcal{X} \subset \mathbb{R}^d \). Suppose \( f \in B_k \). Then there exists a constant \( c > 0 \), such that the query point set generated by Algorithm 5 with a sufficiently small covering radius (fill distance) \( h_X \) achieves a regret bound given by equation (24):

\[
\bar{r}_T \leq B \exp(c \log(h_X)/(2\sqrt{h_X})).
\]

**Remark:** Theorem 6 presents a regret bound for the SE kernel. It attains higher rate w.r.t covering radius \( h_X \) compared with Theorem 5 because functions in RKHS with SE kernel are more smooth than functions in Sobolev space.

We analyze the regret under a more adversarial setting. This relates to a more robust requirement. The regret bounds under a fully adversarial setting when little information is known are summarized in Theorem 7.

**Theorem 7.** Define \( B_k = \{ f : f \in \mathcal{H}_k, \|f\|_{\mathcal{H}_k} \leq B \} \) associated with a shift invariant kernel \( k(x, y) = \Phi(\|x - y\|) \leq 1 \) that decreases w.r.t \( \|x - y\| \). Let \( f_t(x_i^*) = \max_{x \in \mathcal{X}} f_t(x) \) with \( f_t \in B_k \). Then the query point set generated by Algorithm 5 with covering radius (fill distance) \( h_X \) achieves a regret bound as

\[
\bar{r}_T = \min_{t \in [T]} \sup_{f_t \in B_k} \left\{ f_t(x_i^*) - f_t(x_i) \right\} \leq B\sqrt{2 - 2\Phi(h_X)}.
\]
Remark: Theorem 7 gives a fully adversarial bound. Namely, the opponent can choose functions from $B_k$ without the same history. The regret bound decreases with the decrease of the covering radius (fill distance).

Corollary 2. Define $B_k = \{ f : f \in H_k, \| f \|_{H_k} \leq B \}$ associated with squared exponential kernel. Let $f_i(x_t^*) = \max_{x \in X} f_i(x)$ with $f_i \in B_k$. Then the query point set generated by Algorithm 5 with covering radius (fill distance) $h_X$ achieves a regret bound as

$$\bar{r}_T = \min_{t \in \{1, \ldots, T\}} \sup_{f_i \in B_k} \{ f_i(x^*) - f_i(x_t) \} \leq O(h_X).$$

Remark: For a regular grid, $h_X = O(T^{-\frac{1}{d}})$ \cite{30}, we then achieve $\bar{r}_T = O(T^{-\frac{1}{d}})$. Computer search can find a points set with a smaller covering radius than that of a regular grid.

All the adversarial regret bounds discussed above decrease with the decrease of the covering radius. Thus, the point set generated by Algorithm 5 with small covering radius can serve as a good robust initialization for BO.

B Fast Rank-1 Lattice Construction

In this section, we describe the procedure of generating a query points set that has a small covering radius (fill distance). Because minimizing the covering radius of lattice is equivalent to maximizing the packing radius (separate distance) \cite{17}, we generate the query points set through maximizing the packing radius (separate distance) of the rank-1 lattice. An illustration of the rank-1 lattice constructed by Algorithm 6 is given in Fig. 3.

B.1 The rank-1 lattice construction given a base vector

Rank-1 lattice is widely used in the Quasi-Monte Carlo (QMC) literature for integral approximation \cite{17,18}. The lattice points of the rank-1 lattice in $[0, 1]^d$ are generated by a base vector. Given an integer base vector $b \in \mathbb{N}^d$, a lattice set $X$ that consists of $N$ points in $[0, 1]^d$ is constructed as

$$X := \{ x_i := \text{mod}(i \times b, N)/N | i \in \{0, \ldots, N-1\} \},$$

Algorithm 5

Construct Candidate set $X_{T-1}$ with $T-1$ points by minimizing the fill distance (e.g. Algorithm 6).

Query the observations at $X_{T-1}$.

Obtain $m_{T-1}()$ and $\sigma_{T-1}^2()$ via equation (4) and (5).

Choose $x_T = \arg \max_{x \in X} m_{T-1}(x) + B\sigma_{T-1}(x)$

Query the observation $y_T = f(x_T)$ at location $x_T$. 

Figure 3: Lattice Points and Random Points on $[0, 1]^2$
Algorithm 6  Rank-1 Lattice Construction

**Input:** Number of primes $M$, dimension $d$, number of lattice points $N$  
**Output:** Lattice points $X^*$, base vector $b^*$  
Set $p_0 = 2 \times d + 1$, initialize $\rho^* = -1$.  
Construct set $U := \{p | p \in P, p \geq p_0\}$ containing $M$ primes.  
for each $p \in U$ do  
    for $i = 0$ to $p - 1$ do  
        Set $g = \text{mod}(q_i + i, p)$, where $q_i \in \mathbb{R}^{d-1}$ and $q_{ij} = j$.  
        Set $g = \text{round}(N \times \text{mod}(|2\cos(\frac{2\pi g}{p})|, 1))$.  
        Set $b$ as $[1, g]$ by concatenating vector 1 and $g$.  
        Generate lattice $X$ given base vector $b$ as Eq.(27).  
        Calculate the packing radius (separate distance) $\rho_X$ of $X$ as Eq.(29).  
        if $\rho_X > \rho^*$ then  
            Set $b^* = b$ and $\rho^* = \rho_X$.  
        end if  
    end for  
end for  
Generate lattice $X^*$ given base vector $b^*$ as Eq.(27).

Algorithm 7  Rank-1 Lattice Construction with Successive Coordinate Search (SCS)

**Input:** Number of primes $M$, dimension $d$, number of lattice points $N$, number of iteration of SCS search subroutine $T$.  
**Output:** Lattice points $X^*$, base vector $b^*$  
Set $p_0 = 2 \times d + 1$, initialize $\rho^* = -1$.  
Construct set $U := \{p | p \in P, p \geq p_0\}$ containing $M$ primes.  
for each $p \in U$ do  
    for $i = 0$ to $p - 1$ do  
        Set $g = \text{mod}(q_i + i, p)$, where $q_i \in \mathbb{R}^{d-1}$ and $q_{ij} = j$.  
        Set $g = \text{round}(N \times \text{mod}(|2\cos(\frac{2\pi g}{p})|, 1))$.  
        Set $b$ as $[1, g]$ by concatenating vector 1 and $g$.  
        Perform SCS search [1] with $b$ as the initialization base vector to get a better base $\hat{b}$ and $\rho_X$.  
        if $\rho_X > \rho^*$ then  
            Set $b^* = \hat{b}$ and $\rho^* = \rho_X$.  
        end if  
    end for  
end for  
Generate lattice $X^*$ given base vector $b^*$ as Eq.(27).

where $\text{mod}(a, b)$ denotes the component-wise modular function, i.e., $a \% b$. We use $\text{mod}(a, 1)$ to denote the fractional part of number $a$ in this work.

### B.2 The separate distance of a rank-1 lattice

Denote the toroidal distance [11] between two lattice points $y \in [0, 1]^d$ and $z \in [0, 1]^d$ as:

$$
\|y - z\|_T := \sqrt{\sum_{i=1}^{d} (\min(|y_i - z_i|, 1 - |y_i - z_i|))^2}.
$$

(28)

Because the difference (subtraction) between two lattice points is still a lattice point, and a rank-1 lattice has a periodic 1, the packing radius (separate distance) $\rho_X$ of a rank-1 lattice with set $X$ in $[0, 1]^d$ can be calculated as

$$
\rho_X = \min_{x \in X \cap [0, 1]^d} \frac{1}{2} \|x\|_T.
$$

(29)
where \( \|x\|_T \) can be seen as the toroidal distance between \( x \) and \( 0 \). This formulation calculates the packing radius (separate distance) with a time complexity of \( O(Nd) \) rather than \( O(N^2d) \) in pairwise computation.

### B.3 Searching the rank-1 lattice with maximized separate distance

Given the number of primes \( M \), the dimension \( d \), and the number of lattice points \( N \), we try to find the optimal base vector \( b^* \) and its corresponding lattice points \( X^* \) such that the separate distance \( \rho_{X^*} \) is maximized over a candidate set. We adopt the algebra field based construction formula in \cite{14} to construct the base vector of a rank-1 lattice. Instead of using the same form as \cite{14}, we adopt a searching procedure as summarized in Algorithm 6. The main idea is a greedy search starting from a set of \( M \) prime numbers. For each prime number \( p \), it also searches the \( p \) offset from 0 to \( p - 1 \) to construct the possible base vector \( b \) and its corresponding \( X \). After the greedy search procedure, the algorithm returns the optimal base vector \( b^* \) and the lattice points set \( X^* \) that obtains the maximum separate distance. Algorithm 6 can be extended by including successive coordinate search (SCS) \cite{1} as an inner searching procedure. The extended method is summarized in Algorithm 7. This method can achieve superior performance compared to other baselines.

### B.4 Comparison of minimum distance generated by different methods

Table 1: Minimum distance \( (2\rho_{X}) \) of 1,000 lattice points in \([0, 1]^d\) for \( d = 10 \), \( d = 20 \), \( d = 30 \), \( d = 40 \) and \( d = 50 \).

|       | \( d = 10 \) | \( d = 20 \) | \( d = 30 \) | \( d = 40 \) | \( d = 50 \) |
|-------|--------------|--------------|--------------|--------------|--------------|
| Alg 6 | 0.59632      | 1.0051       | 1.3031       | 1.5482       | 1.7571       |
| Korobov | 0.56639 | 0.90139       | 1.0695       | 1.2748       | 1.3987       |
| SCS   | 0.60224      | 1.0000       | 1.2247       | 1.4142       | 1.5811       |
| Alg 7 | \textbf{0.62738} | \textbf{1.0472} | \textbf{1.3620} | \textbf{1.6175} | \textbf{1.8401} |

Table 2: Minimum distance \( (2\rho_{X}) \) of 2,000 lattice points in \([0, 1]^d\) for \( d = 10 \), \( d = 20 \), \( d = 30 \), \( d = 40 \) and \( d = 50 \).

|       | \( d = 10 \) | \( d = 20 \) | \( d = 30 \) | \( d = 40 \) | \( d = 50 \) |
|-------|--------------|--------------|--------------|--------------|--------------|
| Alg 6 | 0.54658      | 0.95561      | 1.2595       | 1.4996       | 1.7097       |
| Korobov | 0.51536 | 0.80039       | 0.96096      | 1.1319       | 1.2506       |
| SCS   | 0.57112      | 0.98420      | 1.2247       | 1.4142       | 1.5811       |
| Alg 7 | \textbf{0.58782} | \textbf{1.0144} | \textbf{1.3221} | \textbf{1.5758} | \textbf{1.8029} |

Table 3: Minimum distance \( (2\rho_{X}) \) of 3,000 lattice points in \([0, 1]^d\) for \( d = 10 \), \( d = 20 \), \( d = 30 \), \( d = 40 \) and \( d = 50 \).

|       | \( d = 10 \) | \( d = 20 \) | \( d = 30 \) | \( d = 40 \) | \( d = 50 \) |
|-------|--------------|--------------|--------------|--------------|--------------|
| Alg 6 | 0.53359      | 0.93051      | 1.2292       | 1.4696       | 1.7009       |
| Korobov | 0.50000 | 0.67185       | 0.82285      | 0.95015      | 1.0623       |
| SCS   | 0.52705      | 0.74536      | 0.91287      | 1.0541       | 1.1785       |
| Alg 7 | \textbf{0.56610} | \textbf{0.98601} | \textbf{1.2979} | \textbf{1.5553} | \textbf{1.7771} |

We evaluate the proposed Algorithm 6 and Algorithm 7 by comparing them with searching in Korobov form \cite{18} and SCS \cite{1}. We fix \( M = 50 \) for Algorithm 6 and Algorithm 7 in all the experiments. The number of iterations of SCS search \cite{1} is set to \( T = 150 \), and number of iterations of SCS search as a subroutine in Algorithm 7 is set to \( T = 3 \).

The minimum distances \( (2\rho_{X}) \) of 1,000 points, 2,000 points and 3,000 points generated by different methods are summarized in Tables 1, 2 and 3 respectively. Algorithm 7 can achieve a larger separate (minimum) distance than other searching methods. This means that Algorithm 7 can generate points set with smaller covering radius (fill distance). Thus, it can generate more robust initialization for BO. Moreover, Algorithm 7 can also be used to generate points for integral approximation on \([0, 1]^d\).
B.5 Comparison between lattice points and random points

The points generated by Algorithm 6 and uniform sampling are presented in Figure 4. We can observe that the points generated by Algorithm 6 cut the domain into several cells. It obtains a smaller covering radius (fill distance) than the random sampling. Thus, it can be used as robust initialization of BO.

C Synthetic Benchmark Test Problems

Synthetic benchmark test problems are listed in Table 4.

D Comparison with Bull’s Non-adaptive Batch Method

Bull [6] presents a non-adaptive batch method with all the query points except one being fixed at the beginning. As mentioned by Bull, this method is not practical. However, Bull [6] does not present an adaptive batch method. We compare our adaptive batch method with Bull’s non-adaptive method on Rosebrock and Ackley functions. The mean values of simple regret over 30 independent runs are presented in Figure 5 which shows that Bull’s non-adaptive method has a very slowly decreasing simple regret.
Table 4: Test functions

| name         | function                                                                 | domain       |
|--------------|---------------------------------------------------------------------------|--------------|
| Rosenbrock   | \[
\sum_{i=1}^{d-1} \left( 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 \right) \] | \([-2, 2]^d\) |
| Nesterov     | \[
\frac{1}{d} |x_1 - 1| + \sum_{i=1}^{d-1} |x_{i+1} - 2|x_i| + 1| \] | \([-2, 2]^d\) |
| Different-Powers | \[
\sum_{i=1}^{d} |x_i|^{2+10^\frac{d-1}{d}} \] | \([-2, 2]^d\) |
| Dixon-Price  | \[
(x_1 - 1)^2 + \sum_{i=2}^{d} i(2x_i^2 - x_{i-1})^2 \] | \([-2, 2]^d\) |
| Ackley       | \[
-20 \exp(-0.2 \sqrt{\frac{1}{d} \sum_{i=1}^{d} x_i^2}) - \exp(\frac{1}{d} \sum_{i=1}^{d} \cos(2\pi x_i)) + 20 + \exp(1) \] | \([-2, 2]^d\) |
| Levy         | \[
+ (w_d - 1)^2 (1 + \sin^2(2\pi w_d)) \] where \(w_i = 1 + (x_i - 1)/4, i \in \{1, ..., d\} \] | \([-10, 10]^d\) |

Figure 5: The mean value of simple regret over 30 runs on Rosenbrock and Ackley function

E  Proof of Theorem 1

Lemma 2. Suppose \(f \in \mathcal{H}_k\) associated with \(k(x, x)\), then \((m_t(x) - f(x))^2 \leq \|f\|^2_{\mathcal{H}_k} \sigma^2_t(x)\)

Proof. Let \(\alpha = K_k^T k_t(x)\). Then we have

\[
(m_t(x) - f(x))^2 = \left( \sum_{i=1}^{t} \alpha_i f(x_i) - f(x) \right)^2 \tag{30}
\]

\[
= \left( \sum_{i=1}^{t} \alpha_i k(x_i, \cdot) - k(x, \cdot), f \right)^2 \tag{31}
\]

\[
\leq \langle f, f \rangle \left( \sum_{i=1}^{t} \alpha_i k(x_i, \cdot) - k(x, \cdot), \sum_{i=1}^{t} \alpha_i k(x_i, \cdot) - k(x, \cdot) \right) \tag{32}
\]

\[
= \|f\|^2_{\mathcal{H}_k} \left\| \sum_{i=1}^{t} \alpha_i k(x_i, \cdot) - k(x, \cdot) \right\|^2_{\mathcal{H}_k} \tag{33}
\]
In addition, we can achieve that
\[
\left\| \sum_{i=1}^{t} \alpha_i k(x_i, \cdot) - k(x, \cdot) \right\|_{H_k}^2 = k(x, x) - 2 \sum_{i=1}^{t} \alpha_i k(x_i, x) + \sum_{i=1}^{t} \sum_{j=1}^{t} \alpha_i \alpha_j k(x_i, x_j)
\]
\[
= k(x, x) - 2 \alpha^T \mathbf{k}_t(x) + \alpha^T \mathbf{K}_t \alpha
\]
\[
= k(x, x) - 2k_t(x)^T \mathbf{K}_t - k_t(x) + k_t(x)^T \mathbf{K}_t - \mathbf{K}_t - \mathbf{k}_t(x)
\]
\[
= k(x, x) - \mathbf{k}_t(x)^T \mathbf{K}_t - \mathbf{k}_t(x)
\]
\[
= \sigma^2_t(x)
\]

Plug (38) into (33), we can attain \((m_t(x) - f(x))^2 \leq \|f\|_{H_k}^2 \sigma^2_t(x)\).

\[\Box\]

**Lemma 3.** \(f(x^*) - f(x_t) \leq 2 \|f\|_{H_k} \sigma_{t-1}(x_t)\).

**Proof.** From Lemma 1 and Algorithm 1, we can achieve that
\[
f(x^*) - f(x_t) \leq m_{t-1}(x^*) + \|f\|_{H_k} \sigma_{t-1}(x^*) - f(x_t)
\]
\[
\leq m_{t-1}(x_t) + \|f\|_{H_k} \sigma_{t-1}(x_t) - f(x_t)
\]
\[
\leq \|f\|_{H_k} \sigma_{t-1}(x_t) + \|f\|_{H_k} \sigma_{t-1}(x_t)
\]
\[
= 2\|f\|_{H_k} \sigma_{t-1}(x_t)
\]

\[\Box\]

**Lemma 4.** Let \(\bar{\sigma}^2_t(x) = k(x, x) - \mathbf{k}_t(x)^T (\sigma^2 \mathbf{I} + \mathbf{K}_t)^{-1} \mathbf{k}_t(x)\). Then \(\sigma^2_t(x) \leq \bar{\sigma}^2_t(x)\).

**Proof.** Since kernel matrix \(\mathbf{K}_t\) is positive semi-definite, it follows that \(\mathbf{K}_t = \mathbf{U}^T \Lambda \mathbf{U}\), where \(\mathbf{U}\) is orthonormal matrix consists of eigenvectors, \(\Lambda\) is a diagonal matrix consists of eigenvalues.

Let \(\beta = \mathbf{U} \mathbf{k}_t(x)\), then we can achieve that
\[
\mathbf{k}_t(x)^T (\sigma^2 \mathbf{I} + \mathbf{K}_t)^{-1} \mathbf{k}_t(x) = \sum_{i=1}^{t} \frac{\beta^2_i}{\sigma^2 + \lambda_i}
\]
\[
\leq \sum_{i=1}^{t} \frac{\beta^2_i}{\lambda_i} = \beta^T \Lambda^{-1} \beta
\]
\[
= \mathbf{k}_t(x)^T \mathbf{U}^T \Lambda^{-1} \mathbf{U} \mathbf{k}_t(x)
\]
\[
= \mathbf{k}_t(x)^T \mathbf{K}_t^{-1} \mathbf{k}_t(x)
\]

It follows that
\[
\sigma^2_t(x) = k(x, x) - \mathbf{k}_t(x)^T \mathbf{K}_t^{-1} \mathbf{k}_t(x)
\]
\[
\leq k(x, x) - \mathbf{k}_t(x)^T (\sigma^2 \mathbf{I} + \mathbf{K}_t)^{-1} \mathbf{k}_t(x)
\]
\[
= \bar{\sigma}^2_t(x)
\]

\[\Box\]

Now, we are ready to prove Theorem 1.
Proof. First, we have
\[
R_T = \sum_{i=1}^{T} f(x^*) - f(x_t) 
\]
(50)
\[
\leq 2\|f\|_{\mathcal{H}_k} \sum_{i=1}^{T} \sigma_{i-1}(x_t) 
\]
(51)
\[
\leq 2\|f\|_{\mathcal{H}_k} \sqrt{T \sum_{i=1}^{T} \sigma_{i-1}^2(x_t)} 
\]
(52)
Since \(s \leq \frac{1}{\log(1+\sigma^{-2})} \log (1 + \sigma^{-2}s)\) for \(s \in [0, 1]\) and \(0 \leq \sigma_{i-1}^2(x_t) \leq k(x, x) \leq 1\) for all \(t \geq 1\), it follows that
\[
\sum_{i=1}^{T} \sigma_{i-1}^2(x_t) \leq \sum_{i=1}^{T} \bar{\sigma}_{i-1}^2(x_t) \leq \frac{1}{\log(1 + \sigma^{-2})} \sum_{i=1}^{T} \log(1 + \sigma^{-2} \sigma_{i-1}^2(x_t)) 
\]
(53)
\[
\leq \frac{2\gamma}{\log(1 + \sigma^{-2})} \quad (54)
\]
Together (52) and (54), we can attain that
\[
R_T \leq 2\|f\|_{\mathcal{H}_k} \sqrt{\frac{2\gamma T}{\log(1 + \sigma^{-2})}} = \|f\|_{\mathcal{H}_k} \sqrt{TC_1 \gamma T} 
\]
(55)
(56)
It follows that \(r_T \leq \frac{R_T}{f} \leq \|f\|_{\mathcal{H}_k} \sqrt{\frac{C_1 \gamma T}{f}}\).

\[\Box\]

**F Proof of Theorem 2**

**Lemma 5.** Suppose \(f \in \mathcal{H}_k\) associated with kernel \(k(x, x)\), then \(\left(\sum_{i=1}^{L} m_i(\bar{x}_i) - \sum_{i=1}^{L} f(\bar{x}_i)\right)^2 \leq \|f\|_{\mathcal{H}_k}^2 (1^T A^1)\), where \(A\) denotes the kernel matrix (covariance matrix) with \(A_{ij} = k(\bar{x}_i, \bar{x}_j) - k_i(\bar{x}_i)^T K^{-1}_i k_i(\bar{x}_j)\).

**Proof.** Let \(\alpha^i = k_i(\bar{x}_i)^T K^{-1}_i\). Then we have
\[
\left(\sum_{i=1}^{L} m_i(\bar{x}_i) - \sum_{i=1}^{L} f(\bar{x}_i)\right)^2 = \left(\sum_{i=1}^{L} \sum_{l=1}^{t} \alpha^i_l f(x_l) - \sum_{i=1}^{L} f(\bar{x}_i)\right)^2 
\]
(57)
\[
= \left(\left(\sum_{i=1}^{L} \sum_{l=1}^{t} \alpha^i_l k(x_l, \cdot) - \sum_{i=1}^{L} k(\bar{x}_i, \cdot), f\right)\right)^2 
\]
(58)
\[
\leq \|f\|_{\mathcal{H}_k}^2 \left\|\sum_{i=1}^{L} \sum_{l=1}^{t} \alpha^i_l k(x_l, \cdot) - \sum_{i=1}^{L} k(\bar{x}_i, \cdot)\right\|_{\mathcal{H}_k}^2 
\]
(59)
In addition, we have
\[
\left\| \sum_{i=1}^{L} \sum_{l=1}^{t} \alpha^*_i k(x_l, \cdot) - \sum_{i=1}^{L} k(\widehat{x}_i, \cdot) \right\|^2_{\mathcal{H}_h}
= \sum_{i=1}^{L} \sum_{j=1}^{L} k(\widehat{x}_i, \widehat{x}_j) - 2 \sum_{i=1}^{L} \sum_{j=1}^{L} \sum_{l=1}^{t} \alpha^*_i k(x_l, \widehat{x}_j) + \sum_{i=1}^{L} \sum_{j=1}^{L} \sum_{l=1}^{t} \alpha^*_i \alpha^*_l k(x_l, x_n)
\]
(60)
\[
= \sum_{i=1}^{L} \sum_{j=1}^{L} k(\widehat{x}_i, \widehat{x}_j) - 2 \sum_{i=1}^{L} \sum_{j=1}^{L} k_t(\widehat{x}_i)^T K_i k_t(\widehat{x}_j) + \sum_{i=1}^{L} \sum_{j=1}^{L} k_t(\widehat{x}_i)^T K_i k_t(\widehat{x}_j)
\]
(61)
\[
= \sum_{i=1}^{L} \sum_{j=1}^{L} A_{ij} = 1^T A 1
\]
(62)
Thus, we obtain \((\sum_{i=1}^{L} m_t(\widehat{x}_i) - \sum_{i=1}^{L} f(\widehat{x}_i))^2 \leq \| f \|^2_{\mathcal{H}_h} (1^T A 1)\).

\[\Box\]

**Lemma 6.** Suppose \( f \in \mathcal{H}_h \) associated with kernel \( k(x, x) \), then
\[
\frac{1}{L} \sum_{i=1}^{L} \left( f(x^*) - f(x_{(n-1)L+i}) \right) \leq m(n-1) L(x^*) + \| f \|^2_{\mathcal{H}_h} \sigma(n-1) L(x^*) - \frac{1}{L} \sum_{i=1}^{L} f(x_{(n-1)L+i})
\]
(63)
\[
= \frac{1}{L} \sum_{i=1}^{L} m(n-1) L(x^*) + \| f \|^2_{\mathcal{H}_h} \left( 2\sqrt{\frac{\text{tr} (\text{cov}_{n-1}(X_n, X_n))}{L}} + \frac{1^T \text{cov}_{n-1}(X_n, X_n) 1}{L^2} \right)
\]
(64)
\[
- \frac{1}{L} \sum_{i=1}^{L} f(x_{(n-1)L+i})
\]
(65)
\[
\leq \frac{1}{L} \sum_{i=1}^{L} m(n-1) L(x_{(n-1)L+i}) + \| f \|^2_{\mathcal{H}_h} \left( 2\sqrt{\frac{\text{tr} (\text{cov}_{n-1}(X_n, X_n))}{L}} - \frac{1^T \text{cov}_{n-1}(X_n, X_n) 1}{L^2} \right)
\]
(66)
\[
- \frac{1}{L} \sum_{i=1}^{L} f(x_{(n-1)L+i})
\]
(67)
\[
= 2\| f \|^2_{\mathcal{H}_h} \sqrt{\frac{\text{tr} (\text{cov}_{n-1}(X_n, X_n))}{L}}
\]
(68)

\[\Box\]

**Lemma 7.** Let \( B_n \) and \( A_n \) be the covariance matrix constructed by Eq.(8) and Eq.(17), respectively. Then \( \text{tr}(B_n) \leq \text{tr}(A_n) \).

19
Finally, we are ready to attain Theorem 2.

**Proof.** It follows directly from Lemma 4.

**Lemma 8.** Let $A_{n-1} = \text{cov}_{n-1}(X_n, X_n)$ as Eq. (17). Denote the spectral norm of matrix $A_{n-1}$ as $\beta_{n-1} = \|A_{n-1}\|_2$. Then $\text{tr}(A_{n-1}) \leq \frac{\beta_{n-1}}{\log(1+\beta_{n-1}\sigma^{-2})} \log \det(I + \sigma^{-2}A_{n-1})$ for any $\sigma \neq 0$.

**Proof.** Since $A_{n-1}$ is a positive semidefinite matrix, we can attain that the eigenvalues of $A_{n-1}$ are all nonnegative. Without loss of generality, assume eigenvalues of $A_{n-1}$ as $0 \leq \lambda_L \leq \ldots \leq \lambda_1$. By the definition of the spectral norm $\beta_{n-1} = \|A_{n-1}\|_2$, we obtain that $0 \leq \lambda_L \leq \ldots \leq \lambda_1 \leq \beta_{n-1}$ since $s \leq \frac{\beta_{n-1}}{\log(1+\beta_{n-1}\sigma^{-2})} \log (1 + \sigma^{-2}s)$ for $s \in [0, \beta_{n-1}]$ and $0 \leq \lambda_i \leq \beta_{n-1}$, $i \in \{1, \ldots, L\}$, we can obtain that inequality (69) holds true for all $i \in \{1, \ldots, L\}$

$$\lambda_i \leq \frac{\beta_{n-1}}{\log(1+\beta_{n-1}\sigma^{-2})} \log \left(1 + \sigma^{-2}\lambda_i\right)$$

(69)

Because $\log \det(I + \sigma^{-2}A_{n-1}) = \sum_{i=1}^L \log \left(1 + \sigma^{-2}\lambda_i\right)$, we can achieve that

$$\text{tr}(A_{n-1}) = \sum_{i=1}^L \lambda_i \leq \frac{\beta_{n-1}}{\log(1+\beta_{n-1}\sigma^{-2})} \log \det(I + \sigma^{-2}A_{n-1})$$

(70)

**Lemma 9.** Let $T = NL$, $K_T$ be the $T \times T$ sized kernel matrix and $I_L$ be the $L \times L$ sized identity matrix. Then $\frac{1}{2} \log \det(I + \sigma^{-2}K_T) = \frac{1}{2} \sum_{n=1}^N \log \det(I_L + \sigma^{-2}A_{n-1})$, where matrix $A_{n-1} = \text{cov}_{n-1}(X_n, X_n)$ as Eq. (17).

**Proof.**

$$\frac{1}{2} \log \det(I_T + \sigma^{-2}K_T) = \frac{1}{2} \log \det(\sigma^2I_T + K_T) - \frac{1}{2} \log \det(\sigma^2I_T)$$

(71)

Using the determinant equation $\det \left( \begin{array}{cc} A & B \\ C & D \end{array} \right) = \det(A) \cdot \det(D - CA^{-1}B)$ in linear algebra, set $A = \sigma^2I_{N-1} + K(\overline{X}_{n-1}, \overline{X}_{n-1})$, $B = K(\overline{X}_{n-1}, X_N)$, $C = B^T$ and $D = \sigma^2I_L + K(X_N, X_N)$ where $\overline{X}_{n-1} = \{x_1, \ldots, x_{(N-1)L}\}$ denote all previous $N-1$ batch of points, $X_N = \{x_{(N-1)L+1}, \ldots, x_N\}$ denote the $N^{th}$ batch of points and $K(\cdot, \cdot)$ denote the kernel matrix constructed by its input. Then, we can achieve that

$$\frac{1}{2} \log \det(\sigma^2I_T + K_T) - \frac{1}{2} \log \det(\sigma^2I_T)$$

(72)

$$= \frac{1}{2} \log \det(\sigma^2I_{(N-1)L} + K(\overline{X}_{N-1}, \overline{X}_{N-1})) + \frac{1}{2} \log \det(\sigma^2I_L + A_{n-1}) - \frac{1}{2} \log \det(\sigma^2I_T)$$

$$= \frac{1}{2} \log \det(\sigma^2I_{(N-1)L} + K(\overline{X}_{N-1}, \overline{X}_{N-1})) + \frac{1}{2} \log \det(I_L + \sigma^{-2}A_{n-1}) - \frac{1}{2} \log \det(\sigma^2I_{(N-1)L})$$

where $A_{n-1} = \text{cov}_{n-1}(X_N, X_N)$ is the covariance matrix between $X_N$ and $X_N$ constructed as Eq. (17).

By induction, we can achieve $\frac{1}{2} \log \det(I_T + \sigma^{-2}K_T) = \frac{1}{2} \sum_{n=1}^N \log \det(I_L + \sigma^{-2}A_{n-1})$.

Finally, we are ready to attain Theorem 2.
Proof. Let covariance matrix $A_{n-1}$ and $B_{n-1}$ be constructed as Eq.(7) and Eq. (8), respectively. Let $\beta_{n-1} = \|A_{n-1}\|_2$. Then, we can achieve that

\[
R_T = \sum_{i=1}^{T} f(x^*) - f(x_i)
\]

(73)

\[
\leq 2\|f\|_H \sum_{n=1}^{N} \sqrt{L \text{tr}(B_{n-1})}
\]

(74)

\[
\leq 2\|f\|_H \sum_{n=1}^{N} \sqrt{L \text{tr}(A_{n-1})}
\]

(75)

\[
\leq 2\|f\|_H \sqrt{NL \sum_{n=1}^{N} \text{tr}(A_{n-1})}
\]

(76)

\[
\leq 2\|f\|_H \sqrt{T \sum_{n=1}^{N} \frac{\beta_{n-1}}{\log (1 + \beta_{n-1}\sigma^2)}} \log \det (I + \sigma^{-2}A_{n-1})
\]

(77)

\[
\leq \|f\|_H \sqrt{TC_2 \sum_{n=1}^{N} \log \det (I + \sigma^{-2}A_{n-1})}
\]

(78)

\[
\leq \|f\|_H \sqrt{TC_2 \gamma_T}
\]

(79)

It follows that $r_T \leq \frac{R_T}{\beta} \leq \|f\|_H \sqrt{\frac{C_2 \gamma_T}{\beta}}$

\[\square\]

G Proof of Theorem 3

Lemma 10. Suppose $h = f + g \in H_+^2$ associated with kernel $k^\sigma(x,y) = k(x,y) + \sigma^2 \delta(x,y)$. Suppose $f \in H_k$ associated with $k$ and $g \in H_{\sigma^2 \delta}$ associated with kernel $\sigma^2 \delta$. Then for $x \neq x_i, i \in \{1,...,t\}$, we have $|\hat{m}_t(x) - f(x)| \leq \|h\|_{H_k} \sigma_t(x) + \left(\|h\|_{H_k} + \|g\|_{H_{\sigma^2 \delta}}\right) \sigma$.

Proof. Let $\alpha = (K_t + \sigma^2 I)^{-1} k_t(x)$. Then we have

\[
(\hat{m}_t(x) - h(x))^2 = \left(\sum_{i=1}^{t} \alpha_i h(x_i) - h(x)\right)^2
\]

(80)

\[
= \left(\sum_{i=1}^{t} \alpha_i k^\sigma(x_i,\cdot) - k^\sigma(x,\cdot), h\right)^2
\]

(81)

\[
\leq \|h\|_{H_k}^2 \left\|\sum_{i=1}^{t} \alpha_i k^\sigma(x_i,\cdot) - k^\sigma(x,\cdot)\right\|_{H_k}^2
\]

(82)

In addition, we can achieve that

\[
\left\|\sum_{i=1}^{t} \alpha_i k^\sigma(x_i,\cdot) - k^\sigma(x,\cdot)\right\|_{H_k}^2 = k^\sigma(x,x) - 2 \sum_{i=1}^{t} \alpha_i k^\sigma(x_i,x) + \sum_{i=1}^{t} \sum_{j=1}^{t} \alpha_i \alpha_j k^\sigma(x_i,x_j)
\]

(83)

\[
= k(x,x) + \sigma^2 - 2 \alpha^T k_t(x) + \alpha^T (K_t + \sigma^2 I) \alpha
\]

(84)

\[
= k(x,x) + \sigma^2 - k_t(x)^T (K_t + \sigma^2 I)^{-1} k_t(x)
\]

(85)
Plug (85) into (82), we can obtain \((\hat{m}_t(x) - h(x))^2 \leq ||h||^2_{\mathcal{H}_k} (\hat{\sigma}_t^2(x) + \sigma^2)\). Thus, we achieve that
\[ |\hat{m}_t(x) - f(x)| \leq |\hat{m}_t(x) - h(x)| + |g(x)| \leq ||h||_{\mathcal{H}_k} \sqrt{\hat{\sigma}_t^2(x) + \sigma^2} + \|g\|_{\mathcal{H}_s} \sigma \leq ||h||_{\mathcal{H}_k} \tilde{\sigma}_t(x) + \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma . \]

Lemma 11. Under same condition as Lemma 10, we have \(f(x^*) - f(x_t) \leq 2 ||h||_{\mathcal{H}_k} \tilde{\sigma}_{t-1}(x_t) + 2 \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma . \)

Proof. From Lemma 10 and Algorithm 3, we can achieve that
\[ f(x^*) - f(x_t) \leq \hat{m}_{t-1}(x^*) + ||h||_{\mathcal{H}_k} \tilde{\sigma}_{t-1}(x^*) + \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma - f(x_t) \leq \hat{m}_{t-1}(x_t) + ||h||_{\mathcal{H}_k} \tilde{\sigma}_{t-1}(x_t) + \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma - f(x_t) \leq 2 ||h||_{\mathcal{H}_k} \tilde{\sigma}_{t-1}(x_t) + 2 \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma . \]

Finally, we are ready to prove Theorem 3.

Proof. First, we have
\[ R_T = \sum_{t=1}^{T} f(x^*) - f(x_t) \leq 2 ||h||_{\mathcal{H}_k} \sum_{t=1}^{T} \tilde{\sigma}_{t-1}(x_t) + 2T \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma \leq 2 ||h||_{\mathcal{H}_k} \sqrt{T \sum_{t=1}^{T} \tilde{\sigma}_{t-1}^2(x_t) + 2T \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma} \]
Since \(s \leq \frac{B}{\log(1 + B\sigma^{-2})} \log(1 + \sigma^{-2}s)\) for \(s \in [0, B]\) and \(0 \leq \tilde{\sigma}_{t-1}^2(x_t) \leq k_\sigma^2(x, x) \leq B\) for all \(t \geq 1\), it follows that
\[ \sum_{t=1}^{T} \tilde{\sigma}_{t-1}^2(x_t) \leq \frac{B}{\log(1 + B\sigma^{-2})} \sum_{t=1}^{T} \log(1 + \sigma^{-2}\tilde{\sigma}_{t-1}^2(x_t)) \leq \frac{2B\gamma_T}{\log(1 + B\sigma^{-2})} \]
Together (94) and (96), we can attain that
\[ R_T \leq 2 ||h||_{\mathcal{H}_k} \sqrt{\frac{2B\gamma_T}{\log(1 + B\sigma^{-2})} + 2T \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma} = ||h||_{\mathcal{H}_k} \sqrt{TC_3\gamma_T} + 2T \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma \]
It follows that \(r_T \leq \frac{R_T}{T} \leq ||h||_{\mathcal{H}_k} \sqrt{\frac{C_3\gamma_T}{T}} + 2 \left(||h||_{\mathcal{H}_k} + \|g\|_{\mathcal{H}_s}z_3\right) \sigma . \)
H Proof of Theorem 4

Lemma 12. Suppose \( h = f + g \in H_k^x \) associated with kernel \( k^\sigma(x, y) = k(x, y) + c^2\delta(x, y) \). Suppose \( f \in H_k \) associated with \( k \) and \( g \in H_{\sigma^2\delta} \) associated with kernel \( \sigma^2\delta \). Suppose \( \tilde{x}_i \neq x_j, i \in \{1, \ldots, L\}, j \in \{1, \ldots, t\} \), then we have

\[
\sum_{i=1}^{L} m_t(\tilde{x}_i) - \sum_{i=1}^{L} f(\tilde{x}_i) \leq \|h\|_{H_k^x} \sqrt{1^T A1 + L^2\sigma^2} + L \|g\|_{H_{\sigma^2\delta}}\sigma
\]  

(99)

where \( A \) denotes the kernel covariance matrix with \( A_{ij} = k(\tilde{x}_i, \tilde{x}_j) - k_i(\tilde{x}_i)^T(K_t + \sigma^2 I)^{-1}k_t(\tilde{x}_j) \)

Remark: Further require \( \tilde{x}_i \neq \tilde{x}_j, \forall i, j \in \{1, \ldots, L\} \) can lead to a tighter bound as

\[
\sum_{i=1}^{L} m_t(\tilde{x}_i) - \sum_{i=1}^{L} f(\tilde{x}_i) \leq \|h\|_{H_k^x} \sqrt{1^T A1 + L\sigma^2} + L \|g\|_{H_{\sigma^2\delta}}\sigma
\]  

(100)

Proof. Let \( \alpha^i = (K_t + \sigma^2 I)^{-1}k_t(\tilde{x}_i) \). Then we have

\[
\left( \sum_{i=1}^{L} \hat{m}_t(\tilde{x}_i) - \sum_{i=1}^{L} h(\tilde{x}_i) \right)^2 = \left( \sum_{i=1}^{L} \sum_{t=1}^{T} \alpha^i_t h(x_t) - \sum_{i=1}^{L} h(\tilde{x}_i) \right)^2 
\]  

(101)

\[
= \left( \left\langle \sum_{i=1}^{L} \sum_{t=1}^{T} \alpha^i_t k^\sigma(x_{t_i}, \cdot) - \sum_{i=1}^{L} k^\sigma(\tilde{x}_i, \cdot), h \right\rangle \right)^2 
\]  

(102)

\[
\leq \|h\|_{H_k^x}^2 \left\| \sum_{i=1}^{L} \sum_{t=1}^{T} \alpha^i_t k^\sigma(x_{t_i}, \cdot) - \sum_{i=1}^{L} k^\sigma(\tilde{x}_i, \cdot) \right\|_{H_k^x}^2 
\]  

(103)

In addition, we have

\[
\left\| \sum_{i=1}^{L} \sum_{t=1}^{T} \alpha^i_t k^\sigma(x_{t_i}, \cdot) - \sum_{i=1}^{L} k^\sigma(\tilde{x}_i, \cdot) \right\|_{H_k^x}^2 
\]  

\[
= \sum_{i=1}^{L} \sum_{j=1}^{L} k^\sigma(\tilde{x}_i, \tilde{x}_j) - 2 \sum_{i=1}^{L} \sum_{j=1}^{L} \sum_{t=1}^{T} \alpha^i_t k^\sigma(x_{t_i}, \tilde{x}_j) + \sum_{i=1}^{L} \sum_{j=1}^{L} \sum_{t=1}^{T} \sum_{n=1}^{n} \alpha^i_n \alpha^j_n k^\sigma(x_{t_i}, x_n) 
\]  

(104)

\[
\leq \sum_{i=1}^{L} \sum_{j=1}^{L} k(\tilde{x}_i, \tilde{x}_j) + L^2\sigma^2 - 2 \sum_{i=1}^{L} \sum_{j=1}^{L} k_i(\tilde{x}_i)^T(K_t + \sigma^2 I)^{-1}k_t(\tilde{x}_j) 
\]  

(105)

\[
= \sum_{i=1}^{L} \sum_{j=1}^{L} A_{ij} + L^2\sigma^2 = 1^T A1 + L^2\sigma^2 
\]  

(106)

Thus, we obtain \( \left( \sum_{i=1}^{L} m_t(\tilde{x}_i) - \sum_{i=1}^{L} h(\tilde{x}_i) \right)^2 \leq \|h\|_{H_k^x}^2 \left( 1^T A1 + L^2\sigma^2 \right) \). Then, we can achieve that

\[
\sum_{i=1}^{L} m_t(\tilde{x}_i) - \sum_{i=1}^{L} f(\tilde{x}_i) \leq \left\| \sum_{i=1}^{L} m_t(\tilde{x}_i) - \sum_{i=1}^{L} h(\tilde{x}_i) \right\| + \sum_{i=1}^{L} \left| g(\tilde{x}_i) \right| 
\]  

(107)

\[
\leq \|h\|_{H_k^x} \sqrt{1^T A1 + L^2\sigma^2} + L \|g\|_{H_{\sigma^2\delta}}\sigma 
\]  

(108)

\( \square \)

Lemma 13. Suppose \( h = f + g \in H_k^x \) associated with kernel \( k^\sigma(x, y) = k(x, y) + c^2\delta(x, y) \). Suppose \( f \in H_k \) associated with \( k \) and \( g \in H_{\sigma^2\delta} \) associated with kernel \( \sigma^2\delta \). Suppose \( x_i \neq x_j \), then
we have

\[ \frac{1}{L} \sum_{i=1}^{L} \left( f(x^*) - f(x_{(n-1)L+i}) \right) \leq 2 \|h\|_{\mathcal{H}_k^*} \sqrt{\frac{\text{tr} \left( \hat{\text{cov}}_{n-1}(X_n, X_n) \right)}{L}} + 2 \left( \|h\|_{\mathcal{H}_k^*} + \|g\|_{\mathcal{H}_{\alpha, 2\delta}} \right) \sigma \]

(109)

where covariance matrix \( \hat{\text{cov}}_{n-1}(X_n, X_n) \) is constructed as Eq.(17) with \( X_n = \{x_{(n-1)L+1}, \ldots, x_{nL}\} \).

**Proof.** Let \( X^* = \{x^*, \ldots, x^*\} \) be \( L \) copies of \( x^* \). Then, we obtain that

\[ \frac{1}{L} \sum_{i=1}^{L} \left( f(x^*) - f(x_{(n-1)L+i}) \right) = f(x^*) - \frac{1}{L} \sum_{i=1}^{L} f(x_{(n-1)L+i}) \]

(110)

\[ \leq \hat{m}_{(n-1)L}(x^*) + \|h\|_{\mathcal{H}_k^*} \hat{\sigma}_{(n-1)L}(x^*) + \left( \|h\|_{\mathcal{H}_k^*} + \|g\|_{\mathcal{H}_{\alpha, 2\delta}} \right) \sigma - \frac{1}{L} \sum_{i=1}^{L} f(x_{(n-1)L+i}) \]

(111)

\[ = \frac{1}{L} \sum_{i=1}^{L} \hat{m}_{(n-1)L}(x^*) + \|h\|_{\mathcal{H}_k^*} \left( 2\sqrt{\frac{\text{tr} \left( \hat{\text{cov}}_{n-1}(X^*, X^*) \right)}{L}} - \sqrt{\frac{1^T \hat{\text{cov}}_{n-1}(X^*, X^*) 1}{L^2}} \right) \]

\[ + \left( \|h\|_{\mathcal{H}_k^*} + \|g\|_{\mathcal{H}_{\alpha, 2\delta}} \right) \sigma - \frac{1}{L} \sum_{i=1}^{L} f(x_{(n-1)L+i}) \]

(112)

\[ \leq \frac{1}{L} \sum_{i=1}^{L} \hat{m}_{(n-1)L}(x_{(n-1)L+i}) + \|h\|_{\mathcal{H}_k^*} \left( 2\sqrt{\frac{\text{tr} \left( \hat{\text{cov}}_{n-1}(X_n, X_n) \right)}{L}} - \sqrt{\frac{1^T \hat{\text{cov}}_{n-1}(X_n, X_n) 1}{L^2}} \right) \]

\[ + \left( \|h\|_{\mathcal{H}_k^*} + \|g\|_{\mathcal{H}_{\alpha, 2\delta}} \right) \sigma - \frac{1}{L} \sum_{i=1}^{L} f(x_{(n-1)L+i}) \]

(113)

\[ \leq \|h\|_{\mathcal{H}_k^*} \left( 2\sqrt{\frac{\text{tr} \left( \hat{\text{cov}}_{n-1}(X_n, X_n) \right)}{L}} - \sqrt{\frac{1^T \hat{\text{cov}}_{n-1}(X_n, X_n) 1}{L^2}} \right) + \left( \|h\|_{\mathcal{H}_k^*} + \|g\|_{\mathcal{H}_{\alpha, 2\delta}} \right) \sigma \]

\[ + \|h\|_{\mathcal{H}_k^*} \sqrt{\frac{1^T \hat{\text{cov}}_{n-1}(X_n, X_n) 1 + L^2 \sigma^2}{L^2}} + \|g\|_{\mathcal{H}_{\alpha, 2\delta}} \sigma \]

(114)

\[ \leq 2\|h\|_{\mathcal{H}_k^*} \frac{\text{tr} \left( \hat{\text{cov}}_{n-1}(X_n, X_n) \right)}{L} + 2 \left( \|h\|_{\mathcal{H}_k^*} + \|g\|_{\mathcal{H}_{\alpha, 2\delta}} \right) \sigma \]

(115)

\[ \square \]

Finally, we are ready to attain Theorem 4.
Applying Theorem 5.4 in [16] with \(h\) together with (122) and (123), absorbing the constant into \(R\). From (116), we know

\[
\tilde{r}_T = \min_{t \leq T} \sup_{f_T \in B_T, f(x) = f_T(x), \forall i \in [t-1]} \{ f_T(x^*) - f_T(x_t) \}
\]

(116)

\[
\leq 2 \|h\|_{H_{k,\sigma}} \sum_{n=1}^{N} \sqrt{L \, tr(A_{n-1})} + 2T \left( \|h\|_{H_{k,\sigma}} + \|g\|_{H_{2,\sigma}} \right) \sigma
\]

(117)

\[
\leq 2 \|h\|_{H_{k,\sigma}} \left[ NL \sum_{n=1}^{N} \sqrt{tr(A_{n-1})} + 2T \left( \|h\|_{H_{k,\sigma}} + \|g\|_{H_{2,\sigma}} \right) \sigma
\]

(118)

\[
\leq 2 \|h\|_{H_{k,\sigma}} \sqrt{T \sum_{n=1}^{N} \frac{\beta_{n-1}}{\log(1 + \beta_{n-1}^{2})}} \log(1 + \sigma^{-2}A_{n-1}) + 2T \left( \|h\|_{H_{k,\sigma}} + \|g\|_{H_{2,\sigma}} \right) \sigma
\]

(119)

\[
\leq \|h\|_{H_{k,\sigma}} \sqrt{TC_4 \sum_{n=1}^{N} \log(1 + \sigma^{-2}A_{n-1})} + 2T \left( \|h\|_{H_{k,\sigma}} + \|g\|_{H_{2,\sigma}} \right) \sigma
\]

(120)

\[
\leq \|h\|_{H_{k,\sigma}} \sqrt{TC_4} + 2T \left( \|h\|_{H_{k,\sigma}} + \|g\|_{H_{2,\sigma}} \right) \sigma
\]

(121)

It follows that \(r_T \leq \frac{R_T}{T} \leq \|h\|_{H_{k,\sigma}} \sqrt{\frac{C_{2,\sigma}}{T}} + 2T \left( \|h\|_{H_{k,\sigma}} + \|g\|_{H_{2,\sigma}} \right) \sigma\)

\(\Box\)

I Proof of Theorem 5

Proof.

\[
\tilde{r}_T = \min_{t \leq T} \sup_{f_T \in B_T, f(x) = f_T(x), \forall i \in [t-1]} \{ f_T(x^*) - f_T(x_t) \}
\]

\[
\leq \sup_{f_T \in B_T, f_T(x) = f_T(x), \forall i \in [t-1]} \{ f_T(x^*) - f_T(x_T) \}
\]

\[
\leq \sup_{f_T \in B_T, f_T(x) = f_T(x), \forall i \in [T-1]} \{ m_{T-1}(x^*) + B\sigma_{T-1}(x^*) - f_T(x_T) \}
\]

\[
\leq \sup_{f_T \in B_T, f_T(x) = f_T(x), \forall i \in [T-1]} \{ m_{T-1}(x_T) + B\sigma_{T-1}(x_T) - f_T(x_T) \}
\]

\[
\leq \sup_{f_T \in B_T, f_T(x) = f_T(x), \forall i \in [T-1]} \{ B\sigma_{T-1}(x_T) + B\sigma_{T-1}(x_T) \}
\]

\[
\leq 2B\sigma_{T-1}(x_T)
\]

(122)

Applying Theorem 5.4 in [16] with \(h_{p,\sigma} \leq h_X\), we can obtain that

\[
\sigma_{T-1}(x_T) \leq Ch_X^{s-d/2}
\]

(123)

Together with (122) and (123), absorbing the constant into \(C\), we can achieve that \(\tilde{r}_T \leq Ch_X^{s-d/2}\)

\(\Box\)

J Proof of Theorem 6

Proof. From , we know \(\tilde{r}_T \leq 2B\sigma_{T-1}(x_T)\). By applying Theorem 11.22 in [30], we can obtain that

\[
2B\sigma_{T-1}(x_T) \leq 2B \exp(c \log(h_X)/(2\sqrt{h_X}))
\]

(124)

It follows that \(\tilde{r}_T \leq 2B \exp(c \log(h_X)/(2\sqrt{h_X}))\).
K Proof of Theorem 7

Proof.

\[ \tilde{r}_T = \min_{t \in [T]} \sup_{f_t \in B_k} \{ f_t(x^*) - f_t(x_t) \} \]

\[ = \min_{t \in [T]} \sup_{f_t \in B_k} \{ \langle k(x^*, \cdot) - k(x_t, \cdot), f_t \rangle \} \]

\[ \leq B \sqrt{\langle k(x^*, \cdot) - k(x_t, \cdot), k(x^* - k(x_t, \cdot) \rangle} \]

\[ \leq \min_{t \in [T]} B \sqrt{(k(x^*, x^*) + k(x_t, x_t) - 2k(x^*, x_t))} \]

\[ \leq B \sqrt{(2 - 2\Phi(h_X))} \] (125)

L Proof of Corollary 2

Proof. From Theorem 7, we can obtain that

\[ \tilde{r}_T = \min_{t \in [T]} \sup_{f_t \in B_k} \{ f_t(x^*) - f_t(x_t) \} \]

\[ \leq B \sqrt{(2 - 2\Phi(h_X))} \]

\[ = B \sqrt{(2 - 2 \exp(-Ch_X^2))} \]

\[ \leq B \sqrt{2Ch_X^2} \]

\[ = B \sqrt{2Ch_X} = O(h_X) \] (126)