Bayesian Optimistic Optimisation with Exponentially Decaying Regret

Hung Tran-The [1] Sunil Gupta [1] Santu Rana [1] Svetha Venkatesh [1]

Abstract

Bayesian optimisation (BO) is a well-known efficient algorithm for finding the global optimum of expensive, black-box functions. The current practical BO algorithms have regret bounds ranging from $\mathcal{O}(\frac{logN}{\sqrt{N}})$ to $\mathcal{O}(e^{-\sqrt{N}})$, where N is the number of evaluations. This paper explores the possibility of improving the regret bound in the noiseless setting by intertwining concepts from BO and tree-based optimistic optimisation which are based on partitioning the search space. We propose the BOO algorithm, a first practical approach which can achieve an exponential regret bound with order $\mathcal{O}(N^{-\sqrt{N}})$ under the assumption that the objective function is sampled from a Gaussian process with a Matérn kernel with smoothness parameter $\nu > 4 + \frac{D}{2}$, where D is the number of dimensions. We perform experiments on optimisation of various synthetic functions and machine learning hyperparameter tuning tasks and show that our algorithm outperforms baselines.

1. Introduction

We consider a global optimisation problem whose goal is to maximise $f(x)$ subject to $x \in \mathcal{X} \subset \mathbb{R}^D$, where D is the number of dimensions and f is an expensive black-box functions that can only be evaluated point-wise. The performance of a global optimisation algorithm is typically evaluated using *simple regret*, which is given as

$$r_N = \sup_{x\in\mathcal{X}} f(x) - \max_{1\leq i\leq N} f(x_i)$$

where x_i is the i-th sample, and N is the number of function evaluations. In this paper, we consider the case that the evaluation of f is noiseless.

Bayesian optimisation (BO) provides an efficient model-based solution for global optimisation. The core idea is to transform a global optimisation problem into a sequence of auxiliary optimisation problems of a surrogate function called the acquisition function. The acquisition function is built using a model of the function through its limited observations and recommends the next function evaluation location. Regret analysis has been done for many existing BO algorithms, and typically the regret is sub-linear following order $\mathcal{O}\left(\sqrt{\frac{logN}{N}}\right)$ (Srinivas et al., 2012; Russo et al., 2018). More recently, (Vakili et al., 2020) have improved this to $\mathcal{O}\left(\frac{1}{\sqrt{N}}\right)$ under the noiseless setting. However, a limitation of BO is that performing such a sequence of auxiliary optimisation problems is expensive.

De Freitas et al. (2012) introduced a Gaussian process (GP) based scheme called δ-cover sampling as an alternative to the acquisition function to trade-off exploration and exploitation. Their method samples the objective function using a finite lattice within a feasible region and doubles the density of points in the lattice at each iteration. However, even in moderate dimensions, their algorithm is impractical since the lattice quickly becomes too large to be sampled in a reasonable amount of time (as pointed out by (Wang et al., 2014; Kawaguchi et al., 2016)).

An alternative practical approach for global optimisation is to consider tree-based optimistic optimisation as in (Munos, 2011; Floudas, 2005). These algorithms partition the search space into finer regions by building a hierarchical tree. The key is to have efficient strategies to identify a set of nodes that may contain the global optimum and then to successively reduce and refine the search space to reach closer to the optimum. As example, DIRECT algorithm (Jones et al., 1993) partitions the search space assuming a global Lipschitz constant. Simultaneous Optimistic Optimisation (SOO) algorithm (Munos, 2011) generalises this by using only local Lipschitz conditions. Under certain assumptions, this algorithm shows the possibility of achieving an exponentially diminishing regret $\mathcal{O}(e^{-\sqrt{N}})$. An additional advantage of such algorithms is that we do not need to perform an auxiliary non-convex optimisation of the acquisition functions as in BO which may be difficult in cases that are high-dimensional (Kandasamy, 2015; Tran-The et al., 2020) or have unbounded search spaces (Tran-The et al., 2020). However, these optimistic algorithms are model-free, that

[1] Applied Artificial Intelligence Institute, Deakin University, Geelong, Australia. Correspondence to: Hung Tran-The <hung.tranthe@deakin.edu.au>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

is, they do not utilise the function observations efficiently.

A natural extension to improve the sample efficiency is to incorporate a model of the objective function into the optimistic strategy. Indeed, works that do this include BaMSOO (Wang et al., 2014) and IMGPO (Kawaguchi et al., 2016). Using a Gaussian process (Rasmussen & Williams, 2005) as a model of the objective function, both algorithms avoid to evaluate the objective function for points known to be sub-optimal with high probability. While BaMSOO has a sub-linear regret bound, IMGPO can achieve an exponential regret bound. However, despite this, IMGPO has not still overcome the worst-case regret bound order $\mathcal{O}(e^{-\sqrt{N}})$ of SOO which does not use any model of the objective function. A natural question is that *is there a practical algorithm for global optimisation that can break this regret bound order* $\mathcal{O}(e^{-\sqrt{N}})$ *under a mild assumption*?

In this paper, we propose a novel approach, which combines the strengths of the tree-based optimistic optimisation methods and Bayesian optimisation to achieve an improved regret bound $\mathcal{O}(N^{-\sqrt{N}})$ in the worst case. Our main contributions are summarised as follows:

- A GP-based optimistic optimisation algorithm using novel partitioning procedure and function sampling;
- Our algorithm has a worst-case regret bound of $\mathcal{O}(N^{-\sqrt{N}})$ in the noiseless setting under the assumption that the objective function is sampled from a Gaussian process with a Matérn kernel with smoothness parameter $\nu > 4 + \frac{D}{2}$, where N is the number of evaluations and D is the number of dimensions. Our algorithm avoids an auxiliary optimisation step at each iteration in BO, and avoids the δ-cover sampling in the approach of De Freitas et al. (2012). To our best knowledge, without using an δ-cover sampling procedure which is impractical, this is the tightest regret bound for BO algorithms;
- To validate our algorithm in practice, we perform experiments on optimisation of various synthetic functions and machine learning hyperparameter tuning tasks and show that our algorithm outperforms baselines.

2. Related Works

In this section, we briefly review some related work additional to the work mentioned in section 1.

In Bayesian optimisation literature, there exist some works that use tree-structure for the search space. While Wang et al. (2018) used a Mondrian tree to partition the search space, a recent work by (Wang et al., 2020) used a dynamic tree via K-means algorithm. However, these works focused on improving BO's performance empirically for large-scale data sets or high-dimensions rather than to improve the regret bound.

There are two viewpoints for BO, Bayesian and non-Bayesian as pointed out by Scarlett et al. (2017). In the non-Bayesian viewpoint, the function is treated as fixed and unknown, and assumed to lie in a reproducing kernel Hilbert space (RKHS). Under this viewpoint, Chowdhury & Gopalan (2017); Janz et al. (2020) provided upper regret bounds while Scarlett et al. (2017) provided lower regret bounds for BO with Matérn kernels. These bounds all are sub-linear. Otherwise, in the Bayesian viewpoint where we assume that the underlying function is random according to a GP, Kawaguchi et al. (2016) showed that BO can obtain an exponential convergence rate. In this paper, we focus on the Bayesian viewpoint and break the regret bound order of IMGPO (Kawaguchi et al., 2016) under some mild assumptions.

The optimistic optimisation methods have also been extended to adapt to different problem settings e.g., noisy setting (Valko et al., 2013; Grill et al., 2015), high dimensional spaces (Qian & Yu, 2016; Al-Dujaili & Suresh, 2017), multi-objective optimisation (Al-Dujaili & Suresh, 2018), or multi-fidelity black-box optimisation (Sen et al., 2018). Some works (Shilton et al., 2017; Theckel Joy et al., 2019) proposed the regret bounds for transfer learning in BO. Our work can be complementary to these works and the integration of our solution with them may be promising to improve their regret bounds.

3. Preliminaries

Bayesian Optimisation The standard BO routine consists of two key steps: estimating the black-box function from observations and maximizing an acquisition function to suggest next function evaluation point.

Gaussian process is a popular choice for the first step. Formally, we have $f(x) \sim \mathcal{GP}(m(x), k(x, x'))$ where $m(x)$ and $k(x, x')$ are the mean and the covariance (or kernel) functions. Given a set of observations $\mathcal{D}_{1:p} = \{x_i, y_i\}_{i=1}^{p}$ under a noiseless observation model $y_i = f(x_i)$, the predictive distribution can be derived as $P(f(x)|\mathcal{D}_{1:p}, x) = \mathcal{N}(\mu_{p+1}(x), \sigma^2_{p+1}(x))$, where $\mu_{p+1}(x) = \mathbf{k}^T K^{-1}\mathbf{y} + m(x)$ and $\sigma^2_{p+1}(x) = k(x, x) - \mathbf{k}^T K^{-1}\mathbf{k}$. In the above expression we define $\mathbf{k} = [k(x, x_1), ..., k(x, x_p)]^T$, $K = [k(x_i, x_j)]_{1 \le i,j \le p}$ and $\mathbf{y} = [y_1, \dots, y_p]$.

Some well-known popular acquisition functions for the second step include upper confidence bound (GP-UCB)(Srinivas et al., 2012), expected improvement (EI) (Bull, 2011), Thompson sampling (TS) (Russo et al., 2018) and predictive entropy search (PES) (Hernández-Lobato et al., 2014). Among them, GP-UCB is given as $\mathcal{U}_p(x) =$

$\mu_p(x) + \beta_p^{1/2}\sigma_p(x)$, where β_p is the parameter balancing between the exploration and exploitation. We will use GP-UCB in our tree expansion scheme to determine the node to be expanded.

In this paper, we focus on the popular class of Matérn kernels for Gaussian process which is defined as

$$k_\nu(x, x') = \frac{\sigma^2}{\Gamma(\nu)2^{\nu-1}}(\frac{||x - x'||_2}{\lambda})^\nu \mathcal{B}_\nu(\frac{||x - x'||_2}{\lambda}),$$

where Γ denotes the Gamma function, $\mathcal{B}_\nu$ denotes the modified Bessel function of the second kind, ν is a parameter controlling the smoothness of the function and σ^2, λ are hyper-parameters of the kernel. We assume that the hyper-parameters are fixed and known in advance. However, our work can also be extended for the unknown hyper-parameters of the Matérn kernel as in (Vakili et al., 2020) (for Bayesian setting). Important special cases of ν include $\nu = \frac{1}{2}$ that corresponds to the exponential kernel and $\nu \to \infty$ that corresponds to the squared exponential kernel. The Matérn kernel is of particular practical significance, since it offers a more suitable set of assumptions for the modeling and optimisation of physical quantities ((Stein, 1999)).

Hierarchical Partition We use the hierarchical partition of the search space as in (Munos, 2011). Given a branch factor m, for any depth h, the search space $\mathcal{X}$ is partitioned into a set of m^h sets $A_{h,i}$ (called cells), where $0 \leq i \leq m^h - 1$. This partitioning is represented as a m-ary tree structure where each cell $A_{h,i}$ corresponds to a node (h, i). A node (h, i) has m children nodes, indexed as $\{(h+1, i_j)\}_{1\leq j \leq m}$. The children nodes $\{(h + 1, i_j), 1 \leq j \leq m\}$ form a partition of the parent's node (h, i). The root of the tree corresponds to the whole domain $\mathcal{X}$. The center of a cell $A_{h,i}$ is denoted by $c_{h,i}$ where f and its upper confidence bound is evaluated.

4. Proposed BOO algorithm

4.1. Motivation

Most of tree-based optimistic optimisation algorithms like SOO, StoSOO (Valko et al., 2013), BaMSOO and IMGPO face a *strict negative correlation* between the branch factor m and the number of tree expansions given a fixed function evaluation budget N. On the one hand, using a larger m makes a tree finer, which helps to reach closer to the optimum. On the other hand, having more expansions in the tree also allows to create finer partitions in multiple regions of the space. Thus both a larger branch factor and a larger number of tree expansions allow an algorithm to get closer to the optimum. However, each time a node is expanded, the algorithms such as SOO, StoSOO spend m function evaluations - one for each of the m children and thus the number of

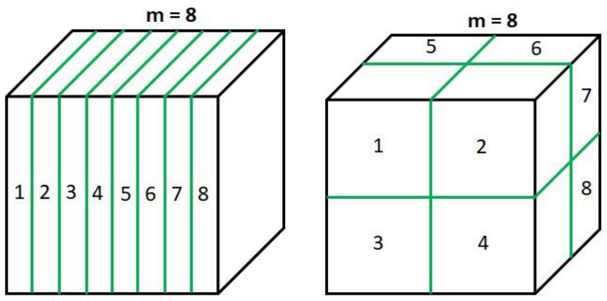

Figure 1. An illustration of partitioning procedure for $m = 8$: (a) SOO methods partition a cell by dividing the longest side into m equal parts; (b) our method sets $m = a^b$ and partitions a cell by dividing $b = 3$ longest sides into $a = 2$ equal parts.

tree expansions is restricted to at most $\lfloor \frac{N}{m} \rfloor$. Thus when m increases, the number of tree expansions decreases. We call this phenomenal the *strict negative correlation* of tree-based optimistic optimisation algorithms.

Using the assumption that the objective function is sampled from a GP prior, BaMSOO and IMGPO reduce this negative correlation by evaluating the function only at the children where the UCB value is greater than the best function value observed thus far (f^+). However, the number of expansions is still tied to the branch factor lying between $\lfloor \frac{N}{m} \rfloor$ and N.

We present a new approach which permits to untie the branch factor m from the number of tree expansions and hence, solves the strict negative correlation of tree-based optimistic optimisation algorithms. By doing so, we can exploit the use of a large m to achieve finer partitions and achieves a regret bound $\mathcal{O}(N^{-\sqrt{N}})$ improving upon current BO algorithms.

4.2. BOO algorithm

Our algorithm is described in Algorithm 1 where we assume that the objective function is a sample from GP as in Bayesian optimisation, however our approach follows the principle of SOO which uses a hierarchical partitioning of the search space $\mathcal{X}$. The main difference of the proposed BOO and previous works lies in the partitioning procedure, the tree expansion mechanism and the function sampling strategy.

Partitioning Procedure Unlike SOO based algorithms including BaMSOO and IMGPO which often divide a cell into m children cells along the longest side of the cell, we use a novel partitioning procedure which exploits the particular decomposition of the branch factor m. Given any $a \geq 2$, $1 \leq b \leq D$ and $m = a^b$, where $a, b, m \in \mathbb{N}$, our partitioning procedure, denoted by $P(m; a, b)$, divides the cell along its b longest dimensions into a new cells (see Figure 1). When $b = 1$ then our procedure becomes simply the

Algorithm 1 The BOO Algorithm

Input: An evaluation budget N and parameters $m, a, b \in \mathbb{N}$.
Initialisation: Set $\mathcal{T}_0 = \{(0,0)\}$ (root node). Set $p = 1$. Sample initial points to build $\mathcal{D}_0$.

```
1:  while True do
2:      Set $v_{max} = -\infty$
3:      for $h = 0$ to min(depth($\mathcal{T}_p$), $h_{max}(p)$) do
4:          Among all leaves $(h, j)$ of depth $h$, select $(h, i) \in \text{argmax}_{(h,j)\in\mathcal{L}} \mathcal{U}_p(c_{h,j})$
5:          if $\mathcal{U}_p(c_{h,i}) \geq v_{max}$ then
6:              Expand node $(h, i)$ by adding $m$ children $(h + 1, i_j)$ to tree $\mathcal{T}_p$, using partitioning procedure $P(m; a, b)$
7:              Evaluate $f(c_{h,i})$
8:              Augment the data $\mathcal{D}_p = \{\mathcal{D}_{p-1}, ((c_{h,i}, f(c_{h,i}))\}$.
9:              Fit the Gaussian process using $\mathcal{D}_p$
10:             Update $v_{max} = \max\{f(c_{h,i}), v_{max}\}$
11:             Update $p = p + 1$
12:             if $p = N$ then
13:                 Return $x(N) = \text{argmax}_{\{c_{h,i})|(c_{h,i},f(c_{h,i}))\in\mathcal{D}_N\}} f(c_{h,i})$
14:             end if
15:         end if
16:     end for
17: end while
```

traditional partitioning procedure as in SOO. When $b = D$, all dimensions of the cell are divided which benefits our algorithm. We will explain this further in our convergence analysis.

Tree Expansion Mechanism The algorithm incrementally builds a tree $\mathcal{T}_p$ starting with a node $\mathcal{T}_0 = \{(0, 0)\}$ for $p = 1...N$, where N is the evaluation budget. At depth h, among all the leaf nodes, denoted by $\mathcal{L}$ of the current tree, the algorithm selects the node with the maximum GP-UCB value, defined as $\mathcal{U}_p(c) = \mu_p(c) + \beta_p^{1/2}\sigma_p(c)$, where $\beta_p^{1/2} = \sqrt{2log(\pi^2 p^3/3\eta)}$ and $\eta \in (0, 1)$. The tree is expanded by adding m children nodes to the selected node. To force the depth of tree after p expansions, we use a function $h_{max}(p)$ which is also a parameter of the algorithm. We note that the algorithm uses GP-UCB acquisition function to determine the candidate node, but it only performs a maximisation on a finite, discrete set comprising the leaf nodes at the depth in consideration.

Function Sampling Strategy Once the node is selected for expansion, unlike previous works that evaluate the objective function at children nodes, we propose to evaluate

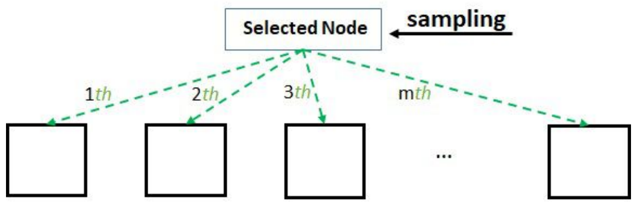

Figure 2. SOO samples the function at all m children nodes while our sampling strategy samples the function only at the parent node (the node selected for expansion). As a result, our strategy requires only one function evaluation irrespective of the value of m.

.

the objective function only at that node without evaluating the function at its children (see Figure 2).

By this sampling scheme, our algorithm allows to untie the branch factor m from the number of tree expansions. As a result, it allows the use of a large m to achieve finer partitions and to reach closer to the optimum. In fact, using a small m as in previous optimistic optimisation methods also reaches finer partitions, however it needs a large number of tree expansions, and thus still needs a large number of evaluations. Consider an example where $m = 2^D$ with $a = 2$ and $b = D$. Using the partitioning procedure $P(m; 2, D)$, our algorithm partitions a node into m cells with the same granularity. To reach the same granularity as our method, SOO algorithm can use the partitioning with $m = 2$ and repeat it D times. However, by this way, SOO always spends 2^D function evaluations while our algorithm only uses one evaluation. BamSOO and IMGPO algorithms have the similar problem although they improve over SOO - they only evaluate the function at nodes c that satisfy the condition $\mathcal{U}(c) > f^+$ (best function value observed thus far) is satisfied. In summary, to reach the same granularity as our method, these algorithms need to spend m' evaluations, where $1 \leq m' \leq 2^D$ depending on the number of nodes c satisfying the condition $\mathcal{U}(c) > f^+$. In contrast, our algorithm only spends one evaluation for all cases regardless of the value of m. Together with our partitioning procedure, we leverage this benefit to improve the regret bound for optimisation.

5. Convergence Analysis

In this section, we theoretically analyse the convergence of our algorithm. We start with assumptions about function f.

5.1. Assumptions

To guarantee the correctness of our algorithm, we use the following assumptions.

Assumption 1. *The function f sampled from $GP(0, k_\nu)$, that is a zero mean GP with a Matérn kernel k_ν with $\nu > 4 + \frac{D}{2}$, where D is the number of dimensions.*

Assumption 2. *The objective function f has a unique global maximum x^*.*

The assumption of a unique maximiser holds with probability one in most non-trivial cases (De Freitas et al., 2012). Under such assumptions, we obtain the following property.

Property 1. *Assume that the function f is sampled from GP($0, k_\nu$) satisfying the Assumption 1 and 2. Then,*

1. $f(x^*) - f(x) \leq L_1||x^* - x||_2^2$ *for every $x \in \mathcal{X}$, for some constant $L_1 > 0$,*
2. $f(x) \leq f(x^*) - L_2||x - x^*||^2$ *for every $x \in \mathcal{B}(x^*, \theta)$ for some constants $L_2, \theta > 0$,*
3. $f(x^*) - max_{x\in\mathcal{X}\setminus\mathcal{B}(x^*,\theta)} f(x) > \epsilon_0$ *for some $\epsilon_0 > 0$.*

We note that all constants L_1, L_2 and ϵ_0 are *unknown*. The Property 1.2 can be considered as the quadratic behavior of the objective function in the neighborhood of the optimum. This property holds for every Matérn kernel with $\nu > 2$ as argued by (De Freitas et al., 2012) and (Wang et al., 2014). The result closest to ours is that of IMGPO (Kawaguchi et al., 2016) that achieves an exponential regret bound $e^{-\sqrt{N}}$. However, their work requires a type of quadratic behavior in the whole search space (as represented in Assumption 2 in their paper) which is quite strong. Compared to it, our assumption is weaker which only requires the quadratic behavior of the function in a neighborhood of the optimum. Despite this, we will show that our algorithm can improve their regret bound.

5.2. Convergence Analysis

For the theoretical guarantee, we follow the principle of the optimism in the face of uncertainty as in (Munos, 2011). The basic idea is to construct the set of expandable nodes at each depth h, called the expansion set. We do this in Section 5.2.1. Quantifying the size of the expansion set is a key step in this principle. We do this in Section 5.2.2. Finally, by using upper bounds on the size of these sets, we derive the regret bounds in Section 5.2.3. **All proofs are provided in the Supplementary Material**.

5.2.1. The expansion set

Definition 1. *Let the expansion set at depth h be the set of all nodes that could be potentially expanded before the optimal node at depth h is selected for expansion in Algorithm 1. Formally,*

$$I_h = \{(h,i) | \exists h \leq p \leq N : \mathcal{U}_p(c_{h,i}) \geq f(x^*) - \delta(h;a,b)\}$$

, where $\delta(h;a,b)$ is defined as

$$\delta(h;a,b) = L_1 D a^{-2\lfloor \frac{bh}{D} \rfloor},$$

and $\mathcal{U}_p(c_{h,i})$ is the upper confidence bound at the center $c_{h,i}$ of node (h,i) after p expansions.

We note that even though this definition uses $\delta(h;a,b)$ that depends on the unknown metric L_1, our BOO algorithm does not need to know this information. The reason we use $\delta(h;a,b) = L_1 D a^{-2\lfloor \frac{bh}{D} \rfloor}$ lies in the following three observations of our partitioning procedure $P(m;a,b)$ in the search space $\mathcal{X}$. We assume here that $\mathcal{X} = [0,1]^D$ (this can always be achieved by scaling).

Lemma 1. *Given a cell $A_{h,i}$ at depth h, we have that*

1. *the longest side of cell $A_{h,i}$ is at most $a^{-\lfloor \frac{bh}{D} \rfloor}$, and*
2. *the smallest side of cell $A_{h,i}$ is at least $a^{-\lceil \frac{bh}{D} \rceil}$.*

Lemma 2. *Given a cell $A_{h,i}$ at depth h, then we have that $sup_{x\in A_{h,i}} ||x - c_{h,i}|| \leq D^{1/2} a^{-\lfloor \frac{bh}{D} \rfloor}$, where $c_{h,i}$ is the center of cell $A_{h,i}$.*

Proof. By Lemma 1, the longest side of a cell at depth h is at most $a^{-\lfloor \frac{bh}{D} \rfloor}$. Therefore, $\sup_{x\in A_{h,i}} ||x - c_{h,i}|| \leq \sqrt{D a^{-2\lfloor \frac{bh}{D} \rfloor}} = D^{1/2} a^{-\lfloor \frac{bh}{D} \rfloor}$. □

We denote a node (h, i^*) as the *optimal node* at depth h if x^* belongs to the cell A_{h,i^*}.

Lemma 3. *At a depth h, we have that $f(c_{h,i^*}) \geq f(x^*) - \delta(h;a,b)$.*

Proof. By Property 1, $f(x^*) - f(c_{h,i^*}) \leq L_1||x^* - c_{h,i^*}||^2$. By Lemma 2, $||x^* - c_{h,i^*}|| \leq D^{1/2} a^{-\lfloor \frac{bh}{D} \rfloor} = \sqrt{\delta(h;a,b)/L_1}$. Thus, $f(x^*) - f(c_{h,i^*}) \leq L_1||x^* - c_{h,i^*}||^2 \leq \delta(h;a,b)$. □

By Lemma 3 and the fact that $\mathcal{U}_p(c_{h,i}) \geq f(c_{h,i})$ with high probability, we have that $\mathcal{U}_p(c_{h,i}) \geq f(x^*) - \delta(h;a,b)$ with high probability. It deduces that the global maximum x^* belongs to the expansion set at every depth with high probability.

The expansion set at a depth h in our approach differs from the ones in works of (Munos, 2011; Wang et al., 2014; Kawaguchi et al., 2016) which is defined as $I_h = \{(h,i)|f(c_{h,i}) \geq f(x^*) - \delta(h;a,b)\}$. More precisely, set I_h of their works is a smaller set than the set I_h in our work defined above because we have $\mathcal{U}_p(c_{h,i}) \geq f(c_{h,i})$ with high probability. This bigger I_h directly might involve unnecessary explorations and therefore, the algorithm may incur higher regret than that of BamSOO and IMGPO. However, we solve this challenge by leveraging our new partitioning procedure $P(m;a,b)$ with $b = D$, and some results from (Kanagawa et al., 2018; Vakili et al., 2020) which are presented in the following section.

5.2.2. An upper bound on the size of the expansion set

To quantify $|I_h|$, we use a concept called the near-optimality dimension as in (Munos, 2011). In our context, we define the near-optimality dimension as follows:

Definition 2. *The near-optimality dimension is defined as the smallest $d \geq 0$ such that there exists $C > 0$ such that for any $\delta(h; a, b)$, the maximal number of disjoint balls the with largest size in a cell at depth h with center in $\mathcal{X}_{\delta(h;a,b)}$ is less than $C(\delta(h; a, b))^d$, where $\mathcal{X}_{\delta(h;a,b)} = \{x \in \mathcal{X} | \mathcal{U}_p(x) \geq f(x^*) - \delta(h; a, b)\}$.*

With partitioning procedure $P(m; a, b)$ where $a = \mathcal{O}(N^{1/D})$ with $b = D$, we will show $d = 0$ (which is equivalent to prove $|I_h| \leq C$) through the following Theorem 1.

Theorem 1 (Bound on Expansion Set Size)**.** *Consider a partitioning procedure $P(m; a, b)$ where $a = \mathcal{O}(N^{1/D})$ and $b = D$. Then there exist constants $N_1 > 0$ and $C > 0$ such that for every for $N \geq N_1$ and $h \geq 2$, we have, $|I_h| \leq C$.*

To prove Theorem 1, we estimate a bound on variance function σ_p in terms of the function $\delta(h-1; a, b)$ through the following lemma.

Lemma 4. *Assuming that node (h, i) at the depth $h \geq 1$ is evaluated at the p-th evaluation, where $p \geq h$. Thus,*

$$\sigma_p(c_{h,i}) \leq C_1(\delta(h-1; a, b))^{\nu/2-D/4},$$

where C_1 is a constant.

This lemma holds by applying some results about the closeness between the samples from a GP (in Bayesian setting) and the elements of an RKHS (in non-Bayesian setting) as in (Kanagawa et al., 2018; Vakili et al., 2020), to the structured search space as in our approach. This technique is novel compared to BaMSOO and IMGPO's ones. We refer to our Supplementary Material in Section 3.

Using this result and the condition $\nu > 4 + D/2$, we achieve a constant bound on the size of set I_h.

5.2.3. Bounding the simple regret

Next we use the upper bound on $|I_h|$ at every depth h to derive a bound on the simple regret r_N.

Let us use h_p^* to denote the depth of the deepest expanded node in the branch containing x^* after p expansions. Similar to the lemma 2 in (Munos, 2011), we can bound the sum of $|I_h|$ as follows.

Lemma 5. *Assume that $f(c) \leq \mathcal{U}(c)$ for all centers c of optimal nodes at all depths $0 \leq h \leq h_{max}(p)$ after p expansions. Then for any depth $0 \leq h \leq h_{max}(p)$, whenever $p \geq h_{max}(p) \sum_{i=0}^{h} |I_i|$, we have $h_p^* \geq h$.*

We use A_p to denote the set of all points evaluated by the algorithm and all centers of optimal nodes of the tree $\mathcal{T}_p$ after p evaluations.

Lemma 6. *Pick a $\eta \in (0, 1)$. Set $\beta_p = 2log(\pi^2 p^3/3\eta)$ and $\mathcal{L}_p(c) = \mu_p(c) - \beta_p^{1/2}\sigma_p(c)$. With probability $1 - \eta$, we have*

$$\mathcal{L}_p(c) \leq f(c) \leq \mathcal{U}_p(c),$$

for every $p \geq 1$ and for every $c \in A_p$.

We now use Lemmas 3-6 to derive a simple regret for the proposed algorithm. Here, the simple regret r_p after p expansions is defined as $r_p = f(x^*) - \max_{1 \leq i \leq p} f(x_i)$, where x_i is the i-th sample.

Theorem 2 (Regret Bound)**.** *Assume that there is a partitioning procedure $P(m; a, b)$ where $a = \mathcal{O}(N^{1/D})$, $b = D$ and $2 \leq m < \sqrt{N} - 1$. Let the depth function $h_{max}(p) = \sqrt{p}$. We consider $m^2 < p \leq N$, and define $h(p)$ as the smallest integer h such that $h \geq \frac{\sqrt{p}-m-1}{C} + 2$, where C is the constant defined by Theorem 1. Pick a $\eta \in (0, 1)$. Then for every $N \geq N_1$, the loss is bounded as*

$$\begin{aligned} r_p &\leq \delta(min\{h(p), \sqrt{p}+1\}; a, b) + \\ &+ 4C_1\beta_p^{1/2}(\delta(min\{h(p)-1, \sqrt{p}\}; a, b))^{\nu/2-D/4}, \end{aligned}$$

with probability $1 - \eta$, where N_1 is the constant defined in Theorem 1, C_1 is the constant defined in lemma 4 and $\beta_N = \sqrt{2log(\pi^2 N^3/3\eta)}$.

Proof. By Theorem 1, the definition of $h(p)$ and the facts that $|I_0| = 1$ and $|I_1| \leq m$, we have

$$\begin{aligned} \sum_{l=0}^{h(p)-1} |I_l| &= |I_0| + |I_1| + (|I_2| + ... + |I|_{h(p)-1}) \\ &\leq 1 + m + C(h(p) - 2) \leq \sqrt{p} \end{aligned}$$

Therefore, $\sum_{l=0}^{h(p)-1} |I_l| \leq \sqrt{p}$. By Lemma 5 when $h(p) - 1 \leq h_{max}(p) = \sqrt{p}$, we have $h_p^* \geq h(p) - 1$. If $h(p) - 1 > \sqrt{p}$ then $h_p^* = h_{max}(p) = \sqrt{p}$ since the BOO algorithm does not expand nodes beyond depth $h_{max}(p)$. Thus, in all cases, $h_p^* \geq \min\{h(p) - 1, \sqrt{p}\}$.

Let (h, j) be the deepest node in $\mathcal{T}_p$ that has been expanded by the algorithm up to p expansions. Thus $h \geq h_p^*$. By Algorithm 1, we only expand a node when its GP-UCB value is larger than v_{max} which is updated at Line 10 of Algorithm 1. Thus, since the node (h, j) has been expanded, its GP-UCB value is at least as high as that of the some node (h_p^*+1, j) at depth h_p^*+1, such that (1) node (h_p^*+1, o) has been evaluated at some p'-th expansion before node (h, j) and (2) $(h_p^* + 1, o) \in \text{argmax}_{(h_p^*+1,i)\in\mathcal{L}} \mathcal{U}_{p'}(c_{h_p^*+1,i})$ (see Line 4 of Algorithm 1).

Thus, by using Lemma 3 and Lemma 6, we can achieve with probability $1-\eta$ that $f(x^*) - U_p(c_{h,j}) \leq \delta(h_p^*+1; a, b) + 2\beta_{p'}^{1/2}\sigma_{p'}(c_{h_p^*+1,o})$.

Further by Lemma 6, we have $U_p(c_{h,j}) = \mu_p(c_{h,j}) + \beta_p^{1/2}\sigma_p(c_{h,j}) = \mathcal{L}_p(c_{h,j}) + 2\beta_p^{1/2}\sigma_p(c_{h,j}) \leq f(c_{h,j}) + 2\beta_p^{1/2}\sigma_p(c_{h,j})$, with a probability $1-\eta$.

Combining these two results, we have $f(x^*) - f(c_{h,j}) \leq \delta(h_p^*+1; a, b) + 2\beta_{p'}^{1/2}\sigma_{p'}(c_{h_p^*+1,o}) + 2\beta_p^{1/2}\sigma_p(c_{h,j})$ with a probability $1-\eta$.

Finally, by using Lemma 4 to bound $\sigma_{p'}(c_{h_p^*+1,o})$ and $\sigma_p(c_{h,j})$ and using the fact that the function $\delta(*; a, b)$ decreases with their depths, we achieve

$$\begin{aligned} r_p &\leq f(x^*) - f(c_{h,j}) \\ &\leq \delta(\min\{h(p), \sqrt{p}+1\}; a, b) + \\ &+ 4C_1\beta_p^{1/2}(\delta(\min\{h(p)-1, \sqrt{p}\}; a, b))^{\nu/2-D/4} \end{aligned}$$

with a probability $1-\eta$. **We provide the complete proof in the Supplementary Material**. □

Finally, we present an improved and simpler expression for the regret bound through the following corollary from Theorem 2.

Corollary 1. *Pick a $\eta \in (0,1)$. Then, there exists a constant $N_2 > 0$ such that for every $N \geq N_2$, the simple regret of the proposed BOO algorithm with the partitioning procedure $P(m; a, b)$ where $a = \lfloor(\frac{\sqrt{N}}{2})^{\frac{1}{D}}\rfloor$, $b = D$, is bounded as*

$$r_N \leq \mathcal{O}(N^{-\sqrt{N}}),$$

with probability $1-\eta$, where N is the number of sampled points.

Remark 1. A detailed expression for the regret bound of Corollary 1 is that $r_N \leq \mathcal{O}(C_1L_1D/2^D N^{-\frac{\sqrt{N}}{CD}+\frac{2}{CD}-\frac{2}{D}})$, where C_1 is a constant (defined in Lemma 4) given L_1 and D, and C is a constant (defined in Theorem 1) given $L1, L_2, \eta$ and ν. This complete formula is extracted from the proof of Corollary 1 in Supplementary Material.

Remark 2. The closest result to ours is the regret bound of IMGPO which has the worst case order $\mathcal{O}(e^{-\sqrt{N}})$. As can be seen, we have improved the regret bound. Our result improves over previous works because we leverage a large value of the branch factor m and our new partitioning procedure with $b = D$ where all dimensions of a cell are divided.

6. Experiments

To evaluate the performance of our BOO algorithm, we performed a set of experiments involving optimisation of three benchmark functions and three real applications. We compared our method against five baselines which have theoretical guarantees: (1) GP-EI (Bull, 2011), (2) GP-UCB (Srinivas et al., 2012), (3) SOO (Munos, 2011), (4) BaMSOO (Wang et al., 2014), (5) IMGPO (Kawaguchi et al., 2016).

Experimental settings All implementations are in Python. For each test function, we repeat the experiments 15 times. We plot the mean and a confidence bound of one standard deviation across all the runs. We used Matérn kernel with $\nu = 4+(D+1)/2$ which satisfies our assumptions, and estimated the kernel hyper-parameters automatically from data using Maximum Likelihood Estimation. All methods using GP (including GP-EI, GP-UCB, BaMSOO, IMGPO and our method) were started from randomly initialised points to train GP. For GP-EI and GP-UCB which follow the standard BO, we used the DIRECT algorithm to maximise the acquisition functions and computed β_t for GP-UCB as suggested in (Srinivas et al., 2012). For tree-based space partitioning methods, we follow their implementations to set the branch factor m. Note that these methods use a small m due to the negative correlation. SOO and BaMSOO use $m = 2$ while IMGPO uses $m = 3$. The depth of search tree $h_{max}(p)$ in SOO and BaMSOO was set to $\sqrt{p}$ as suggested in (Munos, 2011; Wang et al., 2014). The parameter Ξ_n in IMGPO was set to 1.

Table 1. Average CPU time (in seconds) for the experiment with each test function.

Algorithm	Hartmann	Shekel	Schwefel
GP-EI	200.39	740.20	250.79
GP-UCB	880.43	1640.87	180.96
SOO	0.51	0.40	0.11
BaMSOO	39.02	87.62	28.67
IMGPO	23.23	80.53	34.65
BOO	27.21	91.22	41.01

6.1. Optimisation of Benchmark Functions

We first demonstrate the efficiency of our algorithm on standard benchmark functions: Hartmann3 ($D = 3$), Schwefel ($D = 3$) and Shekel ($D = 4$). The evaluation metric is the log distance to the true optimum: $log_{10}(f(x^*) - f^+)$, where f^+ is the best function value sampled so far.

For our BOO algorithm, we choose parameters m, a, b and N as per Corollary 1 which suggests using N so that $a = \mathcal{O}((\frac{\sqrt{N}}{2})^{1/D}) \geq 2$. For Hartmann3 ($D = 3$) and Schwefel ($D = 3$) we use partitioning procedure $P(8; 2, 3)$ with $N = 200$. For Shekel function ($D = 4$), we use $P(16; 2; 4)$ with $N = 800$ so that $(\frac{\sqrt{N}}{2})^{1/D} \approx 2$. We follow Lemma 6 in our theoretical analysis to set $\beta_p = 2log(\pi^2p^3/3\eta)$, with $\eta = 0.05$.

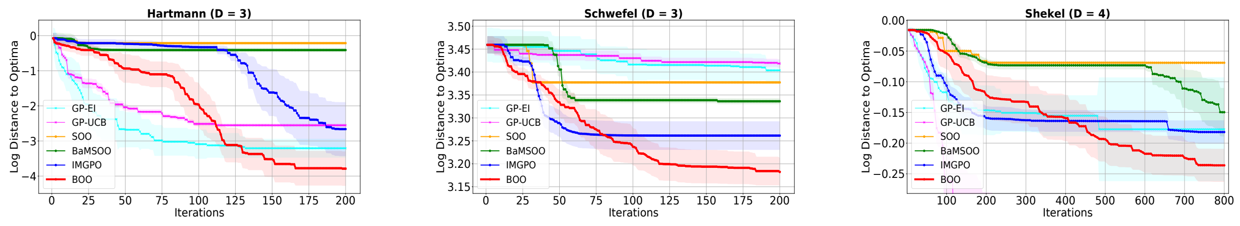

Figure 3. Comparison of methods for Hartmann3 ($D = 3$), Schwefel ($D = 3$), and Shekel ($D = 4$) functions.

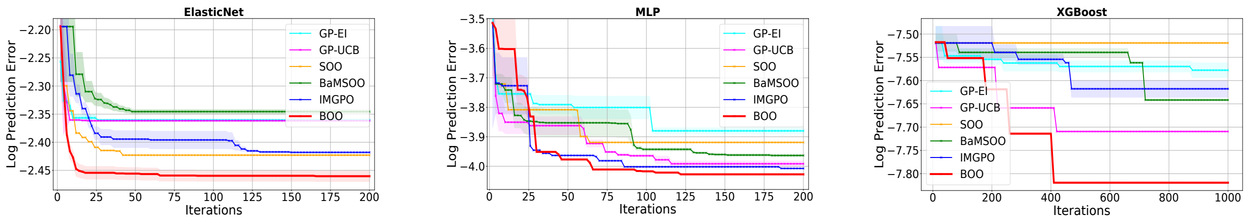

Figure 4. Log prediction error on MNIST dataset for different algorithms ElasticNet, MLP and XGBoost.

Figure 3 shows the performance of our algorithm compared to the baselines. Our method outperforms all baselines for all considered synthetic functions in general with only one exceptional case of Shekel function where GP-UCB performs better our method. Compared to BaMSOO and IMGPO which are tree-based optimisation algorithms, the efficiency of BOO is gained by using a large m and sampling strategies similar to BO (as shown in Section 4.2). Compared to GP-EI and GP-UCB, our algorithm takes advantage of searching a point to be evaluated at each iteration. BOO searches it only in a promising region (as done in Line 4 and 5 in Algorithm 1) rather in a whole search space. Moreover, unlike GP-EI and GP-UCB, BOO avoids the searching by optimisation at each iteration which cannot be obtained sufficiently and accurately given a limited computation budget.

On Computational Effectiveness Our method performs competitively against BaMSOO and IMGPO in terms of computational effectiveness (as shown in Table 1). Our method uses a large value of m and hence it takes slightly more time to compute UCBs of all nodes. It performs slower than IMGPO but much faster than GP-EI, GP-UCB which require the maximisation of the acquisition function in a continuous space.

On Ablation Study between Function Sampling and Partitioning Procedure To show the influence of the proposed function sampling and the proposed partition procedure on BOO's performance, we have performed additional experiments with $m = 64$ (see Figure 5). The left plot shows different partitioning procedures while the function sampling is fixed to our proposed scheme. We can see a good improvement when $b = D$ compared to $b = 1$ case. The right plot compares BaMSOO with our method which uses the proposed function sampling scheme but keeps using BaMSOO's partitioning procedure ($b = 1$, $m = 64$). In this case, we are not able to outperform BaMSOO. However, our result for $b = D$ in the left plot is significantly better than that of BaMSOO. This clearly shows that the effect of partitioning procedure is higher than that of the function sampling.

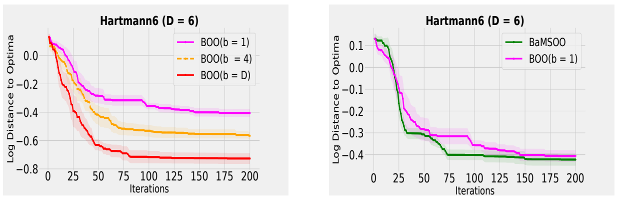

Figure 5. On Ablation Study between Function Sampling and Partitioning Procedure for the Hartmann6 ($D = 6$) function.

6.2. Hyperparameter Tuning for Machine Learning Models

To further validate the performance of our algorithm, we tune hyperparameter tuning of three machine learning models on the MNIST dataset and Skin Segmentation dataset, then plot the log prediction error.

Elastic Net A regression method has the L_1 and L_2 regularisation parameters. We tune w_1 and w_2 where $w_1 > 0$ expresses the magnitude of the regularisation penalty while $w_2 \in [0, 1]$ expresses the ratio between the two penalties. We tune w_1 in the normal space while w_2 is tuned in an exponent space (base 10). The search space is the domain $[0, 1] \times [-3, -1]$. We implement the Elastic net model by using the function SGDClassifier in the scikit-learn package.

Multilayer Perceptron (MLP) We consider a 2-layer MLP with 512 neurons/layer and optimize three hyperparameters: the learning rate l and the L_2 norm regularisation parameters l_{r1} and l_{r2} of the two layers (all tuned in the exponent space (base 10)). The search space is $[-6,-1]^3$. The model is trained with the Adam optimizer in 20 epochs with batch size 128.

Using MNIST dataset, we train the models with this hyperparameter setting using the 55000 patterns and then test the model on the 10000 patterns. The algorithms suggests a new hyperparameter setting based on the prediction accuracy on the test dataset. We set $N = 200$. We use $P(4;2,2)$ for ElasticNet and $P(8;2,3)$ for MLP as per Corollary 1.

As seen in Figure 4, for Elastic Net, our algorithm outperforms the all baselines. For MLP, our algorithm achieves slightly lower prediction errors compared to the baselines because there is a little room to improve where the prediction error of our method for MLP attains 1.8%.

Table 2. Hyperparameters for XGBoost.

Variables	Min	Max
learning rate	0.1	1
max depth	5	15
subsample	0.5	1
colsample	0.1	1
gamma	0	10

XGBoost classification We demonstrate a classification task using XGBoost (Chen & Guestrin, 2016) on a Skin Segmentation dataset [1]. The Skin Segmentation dataset is plit into 15% for training and 85% for testing for a classification problem. There are 5 hyperarameters for XGBoost which is summarized in Table 2. Our proposed BOO is the best solution, outperforming all the baselines by a wide margin.

7. Conclusion

We have presented a first practical algorithm which can achieve an exponential regret bound with tightest order $N^{-\sqrt{N}}$ for Baysian optimisation under the assumption that the objective function is sampled from a Gaussian process with a Matérn kernel with $\nu > 4 + \frac{D}{2}$. Our partitioning procedure and the sampling strategy differ from the existing ones. We have demonstrated the benefits of our algorithm on both synthetic and real world experiments. In the future we plan to extend our work to high dimensions and noisy setting.

[1]https://archive.ics.uci.edu/ml/datasets/skin+segmentation

Acknowledgments

This research was partially funded by the Australian Government through the Australian Research Council (ARC). Prof Venkatesh is the recipient of an ARC Australian Laureate Fellowship (FL170100006).

References

Al-Dujaili, A. and Suresh, S. Embedded bandits for large-scale black-box optimization. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pp. 758–764, 2017.

Al-Dujaili, A. and Suresh, S. Multi-objective simultaneous optimistic optimization. *Inf. Sci.*, 424:159–174, 2018. doi: 10.1016/j.ins.2017.09.066. URL https://doi.org/10.1016/j.ins.2017.09.066.

Bull, A. D. Convergence rates of efficient global optimization algorithms. *J. Mach. Learn. Res.*, 12:2879–2904, November 2011. ISSN 1532-4435.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

Chowdhury, S. R. and Gopalan, A. On kernelized multi-armed bandits. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 844–853, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/chowdhury17a.html.

De Freitas, N., Smola, A. J., and Zoghi, M. Exponential regret bounds for gaussian process bandits with deterministic observations. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pp. 955–962, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Floudas, C. A. *Deterministic Global Optimization: Theory, Methods and (NONCONVEX OPTIMIZATION AND ITS APPLICATIONS Volume 37) (Nonconvex Optimization and Its Applications)*. Springer-Verlag, Berlin, Heidelberg, 2005. ISBN 0792360141.

Grill, J.-B., Valko, M., Munos, R., and Munos, R. Black-box optimization of noisy functions with unknown smoothness. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28*, pp. 667–675. Curran Associates, Inc., 2015.

Hernández-Lobato, J. M., Hoffman, M. W., and Ghahramani, Z. Predictive entropy search for efficient global optimization of black-box functions. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27*, pp. 918–926. Curran Associates, Inc., 2014.

Janz, D., Burt, D., and Gonzalez, J. Bandit optimisation of functions in the matérn kernel rkhs. In Chiappa, S. and Calandra, R. (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2486–2495. PMLR, 26–28 Aug 2020.

Jones, D. R., Perttunen, C. D., and Stuckman, B. E. Lipschitzian optimization without the lipschitz constant. *J. Optim. Theory Appl.*, 79(1):157–181, October 1993. ISSN 0022-3239.

Kanagawa, M., Hennig, P., Sejdinovic, D., and Sriperumbudur, B. K. Gaussian processes and kernel methods: A review on connections and equivalences, 2018.

Kandasamy. High dimensional bayesian optimisation and bandits via additive models. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 295–304. JMLR.org, 2015.

Kawaguchi, K., Kaelbling, L. P., and Lozano-Pérez, T. Bayesian optimization with exponential convergence, 2016. URL https://arxiv.org/abs/1604.01348.

Munos, R. Optimistic optimization of a deterministic function without the knowledge of its smoothness. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 24*, pp. 783–791. Curran Associates, Inc., 2011.

Qian, H. and Yu, Y. Scaling simultaneous optimistic optimization for high-dimensional non-convex functions with low effective dimensions. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pp. 2000–2006, 2016.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

Russo, D., Roy, B. V., Kazerouni, A., Osband, I., and Wen, Z. A tutorial on thompson sampling. *Foundations and Trends in Machine Learning*, 11(1):1–96, 2018. doi: 10.1561/2200000070.

Scarlett, J., Bogunovic, I., and Cevher, V. Lower bounds on regret for noisy Gaussian process bandit optimization. In Kale, S. and Shamir, O. (eds.), *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pp. 1723–1742, Amsterdam, Netherlands, 07–10 Jul 2017. PMLR. URL http://proceedings.mlr.press/v65/scarlett17a.html.

Sen, R., Kandasamy, K., and Shakkottai, S. Multi-fidelity black-box optimization with hierarchical partitions. volume 80 of *Proceedings of Machine Learning Research*, pp. 4538–4547, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Shilton, A., Gupta, S., Rana, S., and Venkatesh, S. Regret Bounds for Transfer Learning in Bayesian Optimisation. In Singh, A. and Zhu, J. (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 307–315, Fort Lauderdale, FL, USA, 20–22 Apr 2017. PMLR.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. W. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Trans. Inf. Theor.*, 58(5):3250–3265, May 2012. ISSN 0018-9448. doi: 10.1109/TIT.2011.2182033. URL http://dx.doi.org/10.1109/TIT.2011.2182033.

Stein, M. L. *Interpolation of spatial data*. Springer Series in Statistics. Springer-Verlag, New York, 1999. ISBN 0-387-98629-4. doi: 10.1007/978-1-4612-1494-6. URL http://dx.doi.org/10.1007/978-1-4612-1494-6. Some theory for Kriging.

Theckel Joy, T., Rana, S., Gupta, S., and Venkatesh, S. A flexible transfer learning framework for bayesian optimization with convergence guarantee. *Expert Systems with Applications*, 115:656–672, 2019. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2018.08.023. URL https://www.sciencedirect.com/science/article/pii/S0957417418305311.

Tran-The, H., Gupta, S., Rana, S., Ha, H., and Venkatesh, S. Sub-linear regret bounds for bayesian optimisation in unknown search spaces. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 16271–16281. Curran Associates, Inc., 2020. URL https://proceedings.

neurips.cc/paper/2020/file/bb073f2855d769be5bf191f6378f7150-Paper.pdf.

Tran-The, H., Gupta, S., Rana, S., and Venkatesh, S. Trading convergence rate with computational budget in high dimensional bayesian optimization. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 2425–2432, 2020.

Vakili, S., Picheny, V., and Durrande, N. Regret bounds for noise-free bayesian optimization, 2020.

Valko, M., Carpentier, A., and Munos, R. Stochastic simultaneous optimistic optimization. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pp. 19–27, 2013.

Wang, L., Fonseca, R., and Tian, Y. Learning search space partition for black-box optimization using monte carlo tree search, 2020.

Wang, Z., Shakibi, B., Jin, L., and de Freitas, N. Bayesian multi-scale optimistic optimization. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, pp. 1005–1014, 2014.

Wang, Z., Gehring, C., Kohli, P., and Jegelka, S. Batched large-scale bayesian optimization in high-dimensional spaces. volume 84 of *Proceedings of Machine Learning Research*, pp. 745–754, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR.