# Multi-Fidelity Black-Box Optimization with Hierarchical Partitions

**Rajat Sen** [1] **Kirthevasan Kandasamy** [2] **Sanjay Shakkottai** [1]

## Abstract

Motivated by settings such as hyper-parameter tuning and physical simulations, we consider the problem of black-box optimization of a function. Multi-fidelity techniques have become popular for applications where exact function evaluations are expensive, but coarse (biased) approximations are available at much lower cost. A canonical example is that of hyper-parameter selection in a learning algorithm. The learning algorithm can be trained for fewer iterations – this results in a lower cost, but its validation error is only coarsely indicative of the same if the algorithm had been trained till completion. We incorporate the multi-fidelity setup into the powerful framework of black-box optimization through hierarchical partitioning. We develop tree-search based multi-fidelity algorithms with theoretical guarantees on simple regret. We finally demonstrate the performance gains of our algorithms on both real and synthetic datasets.

## 1. Introduction

Optimizing a black-box function $f$ over a Euclidean domain $\mathcal{X}$ is a classical problem studied in several disciplines including computer science, mathematics, and operations research. It finds applications in many real world scientific and engineering tasks including scientific experimentation, industrial design, and model selection in statistics and machine learning (Martinez-Cantin et al., 2007; Parkinson et al., 2006; Snoek et al., 2012). Given a budget of $n$ evaluations, an optimization algorithm operates sequentially – at time $t$, it chooses to evaluate $f$ at $x_t$ based on its previous evaluations $\{x_i, f(x_i)\}_{i=1}^{t-1}$. At the end of $n$ evaluations, it makes a recommendation $x(n)$ and its performance is measured by its *simple regret* $R_n$,

$$R_n = \sup_{x \in \mathcal{X}} f(x) - f(x(n)).$$

Our study focuses on applications where exact evaluation of the function $f$ is expensive. As an example, in the case of model selection, training and validating large neural networks can take several hours to days. Similarly, the simulation of an astrophysical process typically takes multiple days even on a cluster of super computers. Traditional methods for black-box optimization are poorly suited for such applications because we need to invest a considerable number of evaluations to optimize $f$. This motivates studying the *multi-fidelity* setting where we have cheaper, but potentially biased approximations to the function (Cutler et al., 2014; Huang et al., 2006a; Kandasamy et al., 2016c; 2017). As an illustration, in a hyper-parameter tuning scenario, the task is to find the best set of hyper-parameters for training a machine learning model. In this setting, the black-box function that needs to be optimized is the validation error after training the learning algorithm to completion (a certain number of maximum iterations) i.e $\mathcal{X}$ represents the allowed set of hyper-parameters while the function represents the validation error after training to completion. However, as training the algorithm till completion is expensive, we may choose to train the learning algorithm for a few iterations at chosen hyper-parameters and then test it on the validation set. These inexpensive validation errors act as the cheap approximations (*fidelities*) to the function value and can indeed provide valuable information regarding the quality of the hyper-parameters.

The multi-fidelity setup for black-box function optimization has been popularly studied in the Bayesian optimization setting (Huang et al., 2006b; Kandasamy et al., 2016b; 2017). However, in this paper we focus on another powerful framework for sequential black-box optimization that works with hierarchical partitioning of the function domain $\mathcal{X}$. These *tree-search* based methods were initially motivated by an empirically successful heuristic UCT (Kocsis & Szepesvári, 2006), which subsequently lead to several theoretically founded algorithms for black-box optimization through hierarchical partitioning (Bubeck et al., 2011; Kleinberg et al., 2008; Munos, 2011; Valko et al., 2013).

In this work, we incorporate cheap approximations or *fideli-*

[1]University of Texas as Austin [2]Carnegie Mellon University. Correspondence to: Rajat Sen <rajat.sen@utexas.edu>.

*Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

*ties* with *tree-search* based methods for black-box optimization. We assume access to a tree-like partitioning of the domain $\mathcal{X}$ similar to (Bubeck et al., 2011; Grill et al., 2015; Munos, 2011). The partitioning of the domain $\mathcal{X}$ is denoted as $\mathcal{P}$ and it contains hierarchical cells $\{\mathcal{P}_{h,i}\}$, where $h$ denotes the height of the cell and $i$ denotes the index. A cell $\mathcal{P}_{h,i}$ at height $h$ has $K$ children $\{\mathcal{P}_{h+1,i_k}\}_{k=1}^{K}$, which are distinct partitions of $\mathcal{P}_{h,i}$. At height 0, there is only one partition $\mathcal{P}_{0,1} = \mathcal{X}$. An example of such an hierarchical partition for $\mathcal{X} = [0,1]$ and $K = 2$ would be: $\mathcal{P}_{0,1} = [0,1]$, $\mathcal{P}_{1,1}, \mathcal{P}_{1,2} = [0,0.5], (0.5,1]$, $\mathcal{P}_{2,1} = [0,0.25]$... and so on. Most of the prior work assume some smoothness property about the function and hierarchical partitioning. We follow a similar path adopting the smoothness assumptions in (Grill et al., 2015). This assumption states that there exists $\nu$ and $\rho \in (0,1)$ such that,

$$\forall h \geq 0, \forall x \in \mathcal{P}_{h,i_h^*}, \qquad f(x) \geq f(x^*) - \nu\rho^h, \qquad (1)$$

where $x^*$ is assumed to be the unique point in $\mathcal{X}$ such that $f(x^*) = \sup_{x\in\mathcal{X}} f(x)$. This assumption basically says that the diameter of the function is bounded for all cells that contain the optima, and that this diameter goes down at a geometric rate with the height of the cell. We also adopt the definition of the well-known *near-optimality* dimension $d(\nu,\rho)$ which restricts the number of cells at height $h$ that contain points close to the optima. This is an important quantity in the analysis of many tree-search based methods (Bubeck et al., 2011; Grill et al., 2015; Munos, 2011; Valko et al., 2013).

In addition we also model that the function can be accessed at a continuous range of fidelities within $\mathcal{Z} = [0,1]$, where $z = 0$ is the cheapest fidelity and $z = 1$ is the most expensive one. For instance, in our hyper-parameter tuning example, $z = 1$ may correspond to training the learning algorithm for 1000 iterations while $z = 0$ represents training the algorithm to 50 iterations. When a function is evaluated at a point $x \in \mathcal{X}$ with a fidelity $z \in \mathcal{Z}$, a value $f_z(x)$ is revealed such that $|f(x) - f_z(x)| < \zeta(z)$ where $\zeta(.)$ is a fixed bias function. The bias function is monotonically decreasing in $z$ with $\zeta(1) = 0$. There is also a cost associated with these evaluations which is captured by a cost function $\lambda : \mathcal{Z} \to \mathbb{R}^+$. The cost function is assumed to be monotonically increasing in $z$. For instance, in hyper-parameter tuning the cost increases linearly with the number of iterations. The objective is to locate a point $x$ such that $f(x)$ is as close as possible to $\sup_{x\in\mathcal{X}} f(x)$ given a finite cost budget $\Lambda$.

The following are the **main contributions** of this work:

(i) We incorporate multiple fidelities/cheap approximations in black-box function optimization through hierarchical partitioning. We propose and analyze two algorithms in this setting. Our first algorithm is known as MFDOO (Algorithm 1) which requires knowledge about the smoothness parameter $(\nu,\rho)$. This algorithm is similar to DOO (Munos, 2011), however it is designed to explore coarser partitions at lower fidelities while exploring finer partitions at higher fidelities, when the algorithm zooms in on a promising area of the function domain. Motivated by recent work (Grill et al., 2015), we also propose a second algorithm MFPDOO (Algorithm 2), which does not require knowledge about the smoothness. This algorithm spawns several instances of MFDOO (Algorithm 1) with carefully selected parameters, at least one of which is bound to perform nearly as well as MFDOO when the smoothness parameters are known.

(ii) We provide simple regret bounds for both of our algorithms, given a fixed cost budget $\Lambda$ for performing evaluations. First we show that when the smoothness parameters are known, MFDOO has simple regret of $\mathcal{O}(\Lambda^{-1/d(\nu,\rho)+1})$ under some conditions on the bias and cost function. Here, $d(\nu,\rho)$ is the *near-optimality dimension* of the function with respect to parameters $(\nu,\rho)$. On the other hand a naive application of DOO (Munos, 2011)[1] only using the highest fidelity $z = 1$ would yield a regret bound of $\mathcal{O}((\Lambda/\lambda(1))^{-1/d(\nu,\rho)})$. We also show that our second algorithm MFPDOO can obtain a simple regret bound of $\mathcal{O}((\Lambda/\log\Lambda)^{-1/d(\nu,\rho)+1})$ even when the smoothness parameters are not known. The precise details about our theoretical guarantees can be found in Section 5.

(iii) Finally, we compare the performance of our algorithms with several state of the art algorithms (Grill et al., 2015; Huang et al., 2006b; Jones et al., 1998; Kandasamy et al., 2016b;b; Srinivas et al., 2009) for black-box optimization in the multi-fidelity setting, on real and synthetic data-sets. We demonstrate that our algorithms outperform the state of the art in most of these experiments.

## 2. Related Work

We build on a line of work on bandits and black-box optimization with hierarchical partitions (Bubeck et al., 2011; Kleinberg et al., 2008; Munos, 2011; Valko et al., 2013). These methods rely on the principle of optimism i.e they build upper bounds on the value of the functions inside different partitions using the already explored points $x_1..., x_t$. Then at time $t + 1$, a point is chosen from the partition that has the highest value of this upper-bound. We closely follow the line of work initiated in (Munos, 2011) that was later extended to noisy function evaluations in (Grill et al., 2015; Valko et al., 2013). In (Munos, 2011) it was assumed that the function follows a local Lipschitz condition with respect to a semi-metric $\ell$, and the diameter of the hierarchical partitions with respect to this semi-metric decrease geometrically with height. Grill et al. (Grill et al., 2015) later merged these

[1]Note that DOO also requires knowledge of the smoothness parameters of the function.

two assumptions into one, by having a single condition that related the smoothness of the function with the hierarchical partition. In this work we adapt the regime of (Grill et al., 2015). However, we also model cheap approximations to the functions through a one-dimensional fidelity space.

Multi-fidelity optimization has had a rich history in many settings (Agarwal et al., 2011; Forrester et al., 2007; Huang et al., 2006a; Klein et al., 2016; Lam et al., 2015; Li et al., 2016; Poloczek et al., 2016; Sabharwal et al., 2015; Zhang & Chaudhuri, 2015), with those that are not application-specific focusing on a Bayesian framework without formal guarantees (we refer to (Kandasamy et al., 2017) for additional discussion). Kandasamy et al. (2016c) propose and analyse a UCB style multi-fidelity algorithm for the $K$-armed bandit setting assuming a finite number of approximations to the $K$ arms. They then extend this work to develop UCB algorithms for black-box optimization under Bayesian Gaussian process assumptions on $f$, both with a finite number of approximations and a continuous spectrum of approximations (Kandasamy et al., 2016a;b; 2017). In all these works, the relation between the approximations and the true function is known and appears in the form of uniform bounds on the approximation or a smoothness assumption arising out of the kernel of the Gaussian process. In our work we merge the multi-fidelity setting with the hierarchical partitions framework.

## 3. Problem Setting

We consider the problem of optimizing a function $f : \mathcal{X} \to \mathbb{R}$ with black-box access at different fidelities. The aim is to locate a point $x$ such that $f(x)$ is as close as possible to $\sup_{x\in\mathcal{X}} f(x)$, given a finite budget for performing evaluations.

We assume that the function can be queried at a continuous range of fidelities in $\mathcal{Z} \triangleq [0,1]$. When the function is queried at a point $x \in \mathcal{X}$ with fidelity $z \in \mathcal{Z}$, a value $f_z(x)$ is revealed. We assume that $|f_z(x) - f(x)| \leq \zeta(z)$, where $\zeta : \mathcal{Z} \to \mathbb{R}^+$ is a known bias function. It is also assumed that a single query at fidelity $z$ incurs a cost $\lambda(z)$, where $\lambda : \mathcal{Z} \to \mathbb{R}^+$ is a known cost function. We assume there is a unique point $x^* \in \mathcal{X}$ at which $\sup_{x\in\mathcal{X}} f(x)$ is achieved.

**Bias and Cost Functions:** The bias function $\zeta$ is assumed to be bounded and monotonically decreasing in $z$. The optimal fidelity $z$ is assumed to have zero bias i.e. $\zeta(1) = 0$. The cost function $\lambda$ is assumed to be bounded and monotonically increasing in $z$.

The multi-fidelity setting is motivated by engineering applications where cheap approximations are available. One promising use case is that of hyper-parameter tuning, where the validation performance of a learning algorithm at different hyper-parameters can be observed. The aim is to locate the best hyper-parameter. In such a setting, cheap approximations are available for instance instead of evaluating the learning algorithm after a maximum of $T$ iterations, one may choose to evaluate it after $t < T$ iterations. In this case $T$ can be mapped to $z = 1$ and $t$ can be mapped to a $z < 1$. The cost function in this setting is proportional to the $O(t)$ computation required. The bias function is monotonically decreasing with $z$, however may not be known exactly in practice. However, prior works in multi-fidelity setup (Kandasamy et al., 2016a;c; Kleinberg et al., 2008) have all assumed access to a known bias function for the theoretical guarantees. Even though we assume the bias function is known in theory, we shall see in our experiments in Section 6 that a simple parametric form of the bias function can be assumed and the parameters can be updated online during the course of our algorithm (similar to (Kandasamy et al., 2017)).

**Simple Regret:** The objective is to locate a point $x$ such that $f(x)$ is as close as possible to $\sup_{x\in\mathcal{X}} f(x)$ given a finite cost budget. Let $\Lambda$ be the total cost budget allowed. Consider an optimization policy that queries a sequence of points $\{x_1, ..., x_{n(\Lambda)}\}$ at fidelities $\{z_1, ..., z_{n(\Lambda)}\}$ respectively and finally returns a recommendation $x_\Lambda$. Our main quantity of interest is the *simple regret* which is defined as,

$$R_\Lambda = \sup_{x\in\mathcal{X}} f(x) - f(x_\Lambda), \tag{2}$$

such that $\sum_{i=1}^{n(\Lambda)} \lambda(z_i) \leq \Lambda$. Note that the simple regret is always measured at the highest fidelity as we are only interested in optimizing the actual function.

### 3.1. Hierarchical Partitions and Assumptions

In this section we define the hierarchical partitions of the domain $\mathcal{X}$ that we assume access to and then provide our technical assumptions about the function and the hierarchical partitions.

**Hierarchical Partitions:** We assume access to a *tree-like* hierarchical partitioning $\mathcal{P} = \{\mathcal{P}_{h,i}\}$ of the domain $\mathcal{X}$, where, $h$ denotes a depth parameter. For any depth $h \geq 0$, the cells $\{\mathcal{P}_{h,i}\}_{1\leq i\leq I_h}$ denote a partitioning of the space $\mathcal{X}$, where $I_h$ is the number of cells at depth $h$. At depth $0$ there is a single cell $\mathcal{P}_{0,1} = \mathcal{X}$. A cell $\mathcal{P}_{h,i}$ can be split into $K$ child nodes at depth $h + 1$. In what follows, querying a cell $\mathcal{P}_{h,i}$ would refer to evaluating the function at a fixed representative point $x_{h,i} \in \mathcal{P}_{h,i}$ at a chosen fidelity. The fixed representative point is usually chosen to be the coordinate wise mid-point for any given cell.

As an *illustrative example* let us consider a hierarchical black-box optimization problem over the domain $\mathcal{X} = [0,1] \times [0,1]$. Let us consider a hierarchical partition of this domain where the cells are of the form $\{x \in \mathcal{X} : b_{1,1} \leq x_1 < b_{1,2}, b_{2,1} \leq x_2 < b_{2,2}\}$. Such a cell will be denoted by the notation $[[b_{1,1}, b_{1,2}], [b_{2,1}, b_{2,2}]]$. Then a hierarchical partition with $K = 2$ starts with the root node $\mathcal{P}_{0,1} = [[0,1],[0,1]]$. This can be further sub-divided into children cells at $h = 1$ given by $\mathcal{P}_{1,1} = [[0, 0.5], [0, 1]]$ and $\mathcal{P}_{1,2} = [[0.5, 1], [0, 1]]$. $\mathcal{P}_{1,2}$ can be further partitioned into $\mathcal{P}_{2,1} = [[0.5, 1], [0, 0.5]]$ and $\mathcal{P}_{2,2} = [[0.5, 1], [0.5, 1]]$ and so on. The fixed representative point for a cell $[[b_{1,1}, b_{1,2}], [b_{2,1}, b_{2,2}]]$ is chosen as the point $[(b_{1,1} + b_{1,2})/2, (b_{2,1} + b_{2,2})/2]$.

Black-box optimization is akin to a needle in a haystack problem without any conditions on the function $f(x)$. Therefore, similar to prior work (Grill et al., 2015) we make the following smoothness assumption which depends on the properties of both the function $f$ and the hierarchical partitioning $\mathcal{P}$.

**Assumption 1** (Smoothness Decay)**.** *There exists $\nu$ and $\rho \in (0, 1)$ such that,*

$$\forall h \geq 0, \forall x \in \mathcal{P}_{h,i_h^*}, \qquad f(x) \geq f(x^*) - \nu\rho^h, \tag{3}$$

where $\mathcal{P}_{h,i_h^*}$ is the unique partition of height $h$ which contains $x^*$.

We also adopt the definition of the *near-optimality-dimension* for parameters $(\nu, \rho)$ from (Grill et al., 2015). This is a quantity that depends on the choice of parameters, the partitioning and the function itself.

**Definition 1.** *The near-optimality dimension of $f$ with respect to parameters $(\nu, \rho)$ is given by,*

$$d(\nu, \rho) \triangleq \inf \Big\{ d' \in \mathbb{R}^+ : \exists C(\nu, \rho), \forall h \geq 0, \\ \mathcal{N}_h(2\nu\rho^h) \leq C(\nu, \rho)\rho^{-d'h} \Big\} \tag{4}$$

*where $\mathcal{N}_h(\epsilon)$ is the number of cells $\mathcal{P}_{h,i}$ such that $\sup_{x \in \mathcal{P}_{h,i}} f(x) \geq f(x^*) - \epsilon$.*

Let $(\nu_*, \rho_*)$ be the parameters with the minimum near optimality dimension $d(\nu^*, \rho^*)$.

**Discussion:** Access to hierarchical partitions have been assumed in a string of previous works on black-box optimization (Bubeck et al., 2011; Grill et al., 2015; Kleinberg et al., 2008; Munos, 2011; Slivkins, 2011; Valko et al., 2013). Many of these prior works assume a semi-metric $\ell$ over the domain $\mathcal{X}$ (Bubeck et al., 2011; Munos, 2011; Valko et al., 2013). In (Bubeck et al., 2011), it is assumed that the function satisfies a weak-Lipschitzness condition. More recent works (Munos, 2011; Valko et al., 2013) have assumed a local-smoothness property w.r.t the metric given by $\forall x \in \mathcal{X}$, $f(x^*) - f(x) \leq \ell(x, x^*)$. However, recently Grill et al. (Grill et al., 2015) have observed that Assumption 1 is sufficient to combine several assumptions about the semi-metric, the function and the hierarchical partitions into one combinatorial condition and similarly have adapted the definition of the *near-optimality* dimension without the semi-metric. It was depicted in (Grill et al., 2015) that prior algorithms like (Bubeck et al., 2011; Munos, 2011; Valko et al., 2013) can be shown to have good regret guarantees with this new set of assumptions, and therefore we adopt these assumptions in our work.

## 4. Algorithms

In this section we present two algorithms for black-box optimization using different fidelities and the hierarchical partitioning provided. In Section 4.1, we provide Algorithm 1 which requires the knowledge of the optimal smoothness decay parameters $(\nu_*, \rho_*)$. Then in Section 4.2, we provide Algorithm 2 that searches for the optimal smoothness by spawning $\mathcal{O}(\log \Lambda)$ instances of Algorithm 1 with a carefully designed sequence of smoothness parameters $(\nu, \rho)$ as arguments.

### 4.1. Algorithm with known $(\nu_*, \rho_*)$

In this section we provide an algorithm which takes as an argument the smoothness parameters $(\nu, \rho)$. We show in Section 5 that if the parameters provided match with the optimal parameters $(\nu_*, \rho_*)$ then the algorithm enjoys strong theoretical guarantees under some conditions on the bias and cost functions $\zeta(z)$ and $\lambda(z)$ respectively.

**Algorithm 1** MFDOO: Multi-Fidelity Deterministic Optimistic Optimization

```
1: Arguments: $(\nu, \rho)$, $\zeta(z)$, $\lambda(z)$, $\mathcal{P}$, $\Lambda$
2: Define $z_h = \zeta^{-1}(\nu\rho^h)$
3: Let $\mathcal{T} = \{(0, 1)\}$ be the tree initialized (root node evaluated with fidelity $z_0$). Set of leaves at time $t$: $\mathcal{L}_t$.
4: Time: $t = 1$; Cost: $C = \lambda(z_0)$.
5: while $C \leq \Lambda$ do
6:     Select the leaf $(h, j) \in \mathcal{L}_t$ with maximum $b_{h,j} \triangleq f_{z_h}(x_{h,j}) + \nu\rho^h + \zeta(z_h)$.
7:     Expand this node; add to $\mathcal{T}_t$ the $K$ children of $(h, j)$.
8:     Evaluate the children at the fidelity level $z_{h+1}$. $t = t + 1$. $C = C + K\lambda(z_{h+1})$.
9: end while
10: Let $h(\Lambda)$ be the height of the tree. Return $x_\Lambda = \arg\max_{(h(\Lambda),i)} f_{z_{h(\Lambda)}}(x_{h(\Lambda),i})$.
```

In Algorithm 1, with some abuse of notation we define for all $h \geq 0$, $z_h = \zeta^{-1}(\nu\rho^h)$ i.e the fidelity at which the bias becomes less than or equal to the smoothness decay parameter at height $h$. All cells at height $h$ are evaluated at the fidelity $z_h$. The intuition is that if $x^*$ belongs to a cell $\mathcal{P}_{h,i^*}$ at height $h$ that has been evaluated, then by Assumption 1 we have that all points in the cell are at least $\nu\rho^h$ optimal. Ideally, beyond this point we would only like to expand leaf nodes that are at least $O(\nu\rho^h)$ optimal, which can only be achieved if the error due to the fidelities is $O(\nu\rho^h)$. At each step, a leaf node with the highest upper bound parameter $b_{h,i}$, is expanded and the children cells are evaluated.

### 4.2. Algorithm without the knowledge of $(\nu_*, \rho_*)$

In this section we describe an algorithm that does not require the optimal parameters $(\nu_*, \rho_*)$. Algorithm 2 just requires $\rho_{max}, \nu_{max}$ which are loose upper-bounds of $\rho_*$ and $\nu_*$ respectively. The algorithm proceeds by spawning $\mathcal{O}(\log \Lambda)$ MFDOO instances with different $(\nu, \rho)$'s which have been carefully designed. Similar ideas were explored in a setting without fidelities in (Grill et al., 2015). In Section 5, we show that Algorithm 2 does almost as well as Algorithm 1 without requiring the optimal parameters as input.

**Algorithm 2** MFPDOO: Multi-Fidelity Parallel Deterministic Optimistic Optimization

```
1: Arguments: $(\nu_{max}, \rho_{max})$, $\zeta(z)$, $\lambda(z)$, $\mathcal{P}$, $\Lambda$
2: Let $N = (1/2)D_{max}\log(\Lambda/\log(\Lambda))$ where $D_{max} = \log K/\log(1/\rho_{max})$
3: for $i = 0$ to $N-1$ do
4:   Spawn MFDOO (Algorithm 1) with parameters $(\nu_{max}, \rho_{max}^{N/(N-i)})$ with budget $(\Lambda - N\lambda(1))/N$
5: end for
6: Let $x_\Lambda^{(i)}$ be the point returned by the $i^{th}$ MFDOO instance for $i \in \{0,..,N-1\}$. Evaluate all $\{x_\Lambda^{(i)}\}_i$ at the $z = 1$. Return the point $x_\Lambda = x_\Lambda^{(i^*)}$ where $i^* = \arg\max_i f(x_\Lambda^{(i)})$.
```

Algorithm 2 proceeds by spawning $N$ different MFDOO instances with the parameters specified in step 4 of the algorithm. We will show in Theorem 2 that at least one of the MFDOO instances will have a performance comparable to Algorithm 1 supplied with parameters $(\nu_*, \rho_*)$ with a budget of $O(\Lambda/N)$. Step 6 of the algorithm obtains the exact value of the points returned by all the MFDOO instances by evaluating them at the highest fidelity, and then chooses the one with the maximum value. This ensures that the highest performing MFDOO instance is selected.

**Remark 1.** *Our algorithms and the theoretical results assume that the bias function $\zeta(.)$ is known. However, in practice we do not assume perfect knowledge about the bias function. We assume a simple parametric form of the bias function and update the parameters online, when the bias assumptions are violated. We provide more details in Section 6 and show that even without this knowledge, the algorithms perform better than other benchmarks.*

It should be noted that the different MFDOO instances created by Algorithm 2 can share information among each other, when multiple instances query the same partition at very similar fidelities. This leads to huge improvements in practice in terms of effectively using the cost budget.

## 5. Theoretical Results

In this section we first prove a general result about the simple regret of Algorithm 1 which assumes access to the optimal parameters $(\nu, \rho)$. This naturally implies a simple regret bound on Algorithm 1 when it is supplied with the parameters $(\nu_*, \rho_*)$ i.e. the ones that have the minimum *near-optimality dimension*. Then we refine these guarantees under some natural conditions on the bias and cost functions. Finally, we show that Algorithm 2 can achieve guarantees close to Algorithm 1 with the optimal parameters, without having access to them.

We first present the following general result about Algorithm 1.

**Theorem 1.** *Let $h'$ be the biggest number $h$ such*

$$\sum_{l=0}^{h} C(\nu, \rho)K\lambda(z_l)\rho^{-d(\nu,\rho)l} \leq \Lambda.$$

*Let $h(\Lambda) = h' + 1$. Then Algorithm 1 run with parameters $(\nu, \rho)$ (s.t $\nu \geq \nu_*, \rho \geq \rho_*$), incurs a simple regret of at most $2\nu\rho^{h(\Lambda)}$ and terminates using a total cost of at most $\Lambda + K\lambda(1)$.*

We defer the proof of Theorem 1 to Section A in the appendix. Note that the guarantee in Theorem 1 is the tightest when the parameters $(\nu_*, \rho_*)$ are supplied as the near optimality dimension $d(\nu_*, \rho_*)$ is the lowest.

Now, we impose some natural conditions on the cost and bias functions. We provide more specialized versions of the guarantees in Theorem 1 under these two conditions *separately*, which are described below.

**Assumption 2.** *We assume that $\zeta(.)$ and $\lambda(.)$ are such that $\lambda(z_h^*) \leq \min\{\beta h, \Lambda(1)\}$ for some positive constant $\beta$. Here, $z_h^* = \zeta^{-1}(\nu_*\rho_*^h)$.*

**Motivation:** The above assumption is motivated by the following hyper-parameter tuning scenario. Consider training a learning algorithm with a particular hyper-parameter that involves optimizing a strongly convex and smooth function with gradient descent. Let the fidelity denote a rescaled version of the number of steps in gradient descent $n$. We assume that at the optimal fidelity ($N$ steps) we reach the optimal value of the function up to an error of $\epsilon_*$. Let $z_n = n/N$. At fidelity $z_n$ the error decays to $\zeta(z_n) = O(r^n)$ for some $r \in (0, 1)$. The cost incurred is linear in the number of steps say $\lambda(z_n) = sn$ for $s > 0$. In this setting it can be shown that if $\zeta(z_n) \sim \nu_*\rho_*^h$, then $n = O(h)$ and therefore $\lambda(z_n) = O(h)$.

The second assumption under which we provide specialized guarantees is as follows.

**Assumption 3.** *We assume that $\zeta(.)$ and $\lambda(.)$ are such that $\lambda(z_h^*) \leq \min\{\gamma^{-h}, \Lambda(1)\}$ for some constant $\gamma \in (\rho, 1)$. Here, $z_h^* = \zeta^{-1}(\nu_* \rho_*^h)$.*

**Motivation:** Assumption 3 is motivated by a similar hyper-parameter tuning scenario as above. Consider training a learning algorithm with a particular hyper-parameter that involves optimizing a smooth convex function with accelerated gradient descent. Let the fidelity denote a rescaled version of the number of steps in gradient descent $n$ as above. At fidelity $z_n$ the error decays to $\zeta(z_n) = O(1/n^2)$. The cost incurred is linear in the number of steps say $\lambda(z_n) = sn$ for $s > 0$. In this setting it can be shown that if $\zeta(z_n) \sim \nu_* \rho_*^h$, then $n = O(\gamma^{-h})$ for $\gamma = O(\sqrt{\rho})$.

We are now at a position to introduce a specialized corollary of Theorem 1.

**Corollary 1.** *Algorithm 1 with parameters $(\nu, \rho)$ (s.t $\nu \geq \nu_*, \rho \geq \rho_*$) run with a total budget of $\Lambda$ terminates with a total cost of at most $\Lambda + K\lambda(1)$ and has the following properties:*

*(i) Under Assumption 2: $R_\Lambda \leq 2\nu \left(\frac{C(\nu,\rho)K\beta}{\Lambda(1-\rho^{d(\nu,\rho)})}\right)^{\frac{1}{d(\nu,\rho)+\epsilon}}$ for some small $\epsilon > 0$, provided $\Lambda$ is large enough.*

*(ii) Under Assumption 3:*

$$R_\Lambda \leq 2\frac{\nu}{\rho}\left(\frac{2C(\nu,\rho)K}{\Lambda(\gamma^{-1}\rho^{-d(\nu,\rho)}-1)}\right)^{\frac{1}{d(\nu,\rho)+1}}.$$

**Comparison with DOO (Munos, 2011):** The above result can be directly compared to DOO (Munos, 2011) which is in the noiseless black-box optimization regime, without access to fidelities. The simple regret of DOO under the same assumptions would scale as $\mathcal{O}\left((\Lambda/\lambda(1))^{-1/d(\nu_*,\rho_*)}\right)$ when all the evaluations are performed at the highest fidelity. In contrast our bounds under Assumption 2 scales as $\mathcal{O}\left((\Lambda/\beta)^{-1/(d(\nu_*,\rho_*)+\epsilon)}\right)$, where $\epsilon$ is a constant close to zero. Note that $\lambda(z_h^*) \leq \lambda(1)$, and therefore $\beta = \lambda(1)$ trivially satisfies the inequality in Assumption 2. Typically, $\beta$ is expected to be much less as compared to the highest fidelity cost $\lambda(1)$. For example in our hyper-parameter tuning example where the fidelity is the number of iterations (a maximum of $N$ iterations), $\beta$ is a small constant (see the discussion on Assumption 2), while $\lambda(1)$ can be $O(N)$. This can lead to significant gains in simple regret as we show in our empirical results in Section 6. Similarly, under Assumption 3 our simple regret scales as $\mathcal{O}\left(\Lambda^{-1/(d(\nu_*,\rho_*)+1)}\right)$, which can be much better than that of DOO (Munos, 2011) as the total budget is not divided by $\lambda(1)$.

Now, we will provide one of our main results which states that Algorithm 2 can recover simple regret bounds which are close to that of Algorithm 1 even when the optimal smoothness parameters are not known.

**Theorem 2.** *Algorithm 2 when run with upper-bounds $\nu_{max}$ and $\rho_{max}$ with a total cost budget of $\Lambda$ terminates after using up a cost of at most $\Lambda + \mathcal{O}(K\lambda(1)\log\Lambda)$ and has the following regret guarantees:*

*(i) Under Assumption 2 the simple regret is* $\mathcal{O}\left((\nu_{max}/\nu_*)^{\frac{D_{max}}{\epsilon+d(\nu_*,\rho_*)}} \times \left(\frac{2\Lambda}{K\beta D_{max}\log(\Lambda/\log\Lambda)} - \frac{\lambda(1)}{K\beta}\right)^{-\frac{1}{\epsilon+d(\nu_*,\rho_*)}}\right)$

*(ii) Under Assumption 3 if $\gamma \geq \rho_{max}$ the simple regret is* $\mathcal{O}\left((\nu_{max}/\nu_*)^{\frac{2D_{max}}{1+d(\nu_*,\rho_*)}} \times \left(\frac{2\Lambda}{KD_{max}\log(\Lambda/\log\Lambda)} - \frac{\lambda(1)}{K}\right)^{-\frac{1}{1+d(\nu_*,\rho_*)}}\right)$.

We defer the proof of this theorem to Appendix C.

**Comparison with POO (Grill et al., 2015):** It is worthwhile to compare our result with that of POO (Grill et al., 2015) which uses only the highest fidelity. It should be noted that POO is in a noisy setting, which gives rise to extra polylog factors in the bounds. However, ignoring polylog factors the simple regret bound of POO would scale as $\mathcal{O}\left((\Lambda/(\log(\Lambda/\lambda(1)) * \lambda(1)))^{-1/(d(\nu_*,\rho_*)+2)}\right)$. In contrast our bounds scale as $\mathcal{O}\left((\Lambda/(\beta\log(\Lambda)))^{-1/(d(\nu_*,\rho_*)+\epsilon)}\right)$ and $\mathcal{O}\left((\Lambda/\log(\Lambda))^{-1/(d(\nu_*,\rho_*)+1)}\right)$ under assumptions 2 and 3 respectively. This can lead to much better performance at the same cost budget. We demonstrate this in our empirical results in Section 6.

## 6. Empirical Results

In this section we provide empirical results on synthetic and real datasets. We compare our algorithm with the following related works: (i) BOCA (Kandasamy et al., 2017) which is a multi-fidelity Gaussian Process (GP) based algorithm that can handle continuous fidelity spaces, (ii) MF-GP-UCB (Kandasamy et al., 2016c) which is a GP based multi-fidelity method that can handle finite fidelities, (iii) GP-EI criterion in bayesian optimization (Jones et al., 1998), (iv) MF-SKO, the multi-fidelity sequential kriging optimisation method (Huang et al., 2006b), (v) GP-UCB (Srinivas et al., 2009) and (vi) MFPDOO($z = 1$) which is a version of our algorithm that uses only the highest fidelity; this is very similar to POO (Grill et al., 2015) but in a noiseless setting. This algorithm is referred to as PDOO in the figures, which is essentially DOO (Munos, 2011) with the smoothness parameters tuned according to the scheme in POO (Grill et al., 2015).

For our theoretical guarantees the bias function $\zeta$ is assumed to be known. However, in practice we assume a parametric

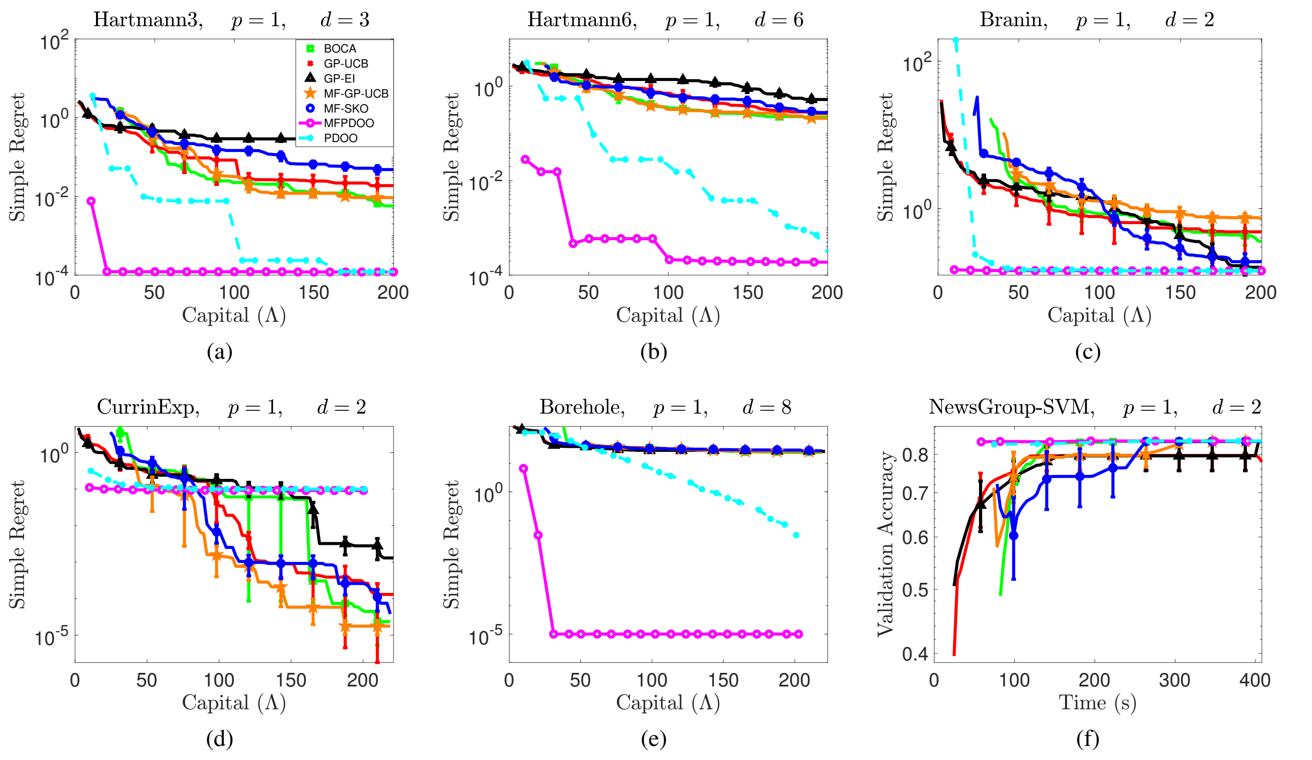

*Figure 1.* We compare all the algorithms in both synthetic and real settings. Note that the common legend for all the figures are provided in Fig. (a) in the interest of space. Figs. (a) to (e) consists of simulation experiments with standard benchmark functions in global optimization. The function name and the dimensions of the fidelity space and the domain are mentioned on top of the individual figures. The x-axis plots the total cost budget $\Lambda$ in multiples of the highest fidelity cost $\lambda(1)$. The y-axis denotes the simple regret in a logarithmic scale. Fig (f) represents a hyper-parameter tuning application on the 20 news group dataset, where the aim is to tune two hyper-parameters of an SVM classifier. The metric is 5-folds cross-validation accuracy. The fidelities used are the number of samples used in estimating the cv accuracy, with the highest fidelity being 5000 samples. The x-axis plots the time taken for the experiments to run.

form for the bias function that is $\zeta(z) = c(1 - z)$ where $c$ is initially set to a very small constant like $0.001$ in our experiments. The nature of Algorithm 2 is such that the same cells are queried at different fidelities by the different MFDOO instances spawned. If a cell is queried at two different fidelities $z_1$ and $z_2$ and the function values obtained are $f_1$ and $f_2$, then we update $c$ to $2c$ whenever $c|z_1 - z_2| < |f_1 - f_2|$. The above update is only performed if $|z_1 - z_2|$ is greater than a specified threshold ($0.0001$ in our experiments). The hierarchical partitioning is performed according to a scheme similar to that of the DIRECT algorithm (Finkel, 2003), where each time a cell is split into $K$ children the dimension that has the biggest width is split into $K$ regions. We set $K = 2$ in all our experiments. Now, we will present the results of our synthetic experiments.

Our implementation can be found at https://github.com/rajatsen91/MFTREE_DET.

## 6.1. Synthetic Experiments

We evaluate all the algorithms on standard benchmark functions used in global optimization. The functions have been modified to incorporate the fidelity space $\mathcal{Z} = [0, 1]$. The setup followed is identical to the one in (Kandasamy et al., 2017), except that we only work in a one dimensional fidelity space. Also, we perform our experiments in a noiseless setting and therefore no Gaussian noise is added to the function evaluations, unlike in (Kandasamy et al., 2017). Note that MF-GP-UCB and MF-SKO are finite fidelity methods. The approximations for these methods are obtained at $z = 0.333$ and $z = 0.667$. We provide more details about the synthetic functions and the fidelities in Appendix D. Our experiments are performed under the deterministic setting, where no noise is added to the approximations. However, several of the algorithms that we compare to have a randomized component. For these algorithms, the results are averaged over 10 experiments and the corresponding error bars are shown. In our algorithm we set the number of MFDOO instances spawned to be $N = 0.1D_{max}\log(\Lambda/\lambda(1))$, given a total budget $\Lambda$. We set $\rho_{max} = 0.95$ and $\nu_{max} = 2.0$.

The results of the synthetic experiments are shown in Figure 1(a)-(e), where the title of each figure shows the name of the function, the dimension of the domain $(d)$ and the dimension of the fidelity space $(p)$. We have $p = 1$ in all our experiments. It can be observed the tree based methods outperform the other algorithms by a large margin, except in the experiments with the CurinExp function (Fig. 1c). Tree-based methods can handle higher dimensions better, as we can see in the Hartman6 (Fig. 1b) and Borehole (Fig. 1e) function experiments. Note that MFPDOO also beats PDOO by a large margin which only uses the highest fidelity. PDOO is essentially DOO (Munos, 2011) where the smoothness decay parameters are tuned according to the scheme in (Grill et al., 2015). MFPDOO can effectively explore the space at cheaper fidelities and then expend the higher fidelities in promising regions of the domain, unlike PDOO.

### 6.2. Tuning SVM for News Group Classification

In this section we describe our experiments that involve tuning hyper-parameters for text classification. For this purpose we use a subset of the 20 news group dataset (Joachims, 1996). All the algorithms are used for tuning two hyper-parameters: (i) the regularization penalty and (ii) the temperature of the rbf kernel both in the range of $[10^{-2}, 10^{3}]$. For our experiments, we use the scikit-learn implementation of SVM classifier and also the inbuilt KFold function for cross-validation. The bag of words in each of the text document is converted into tf-idf features before applying the classification models. We use a one-dimensional fidelity space, where the fidelity denotes the number of samples used to obtain 5-fold cross-validation accuracy. $z = 1$ corresponds to $5000$ samples which is the maximum number of samples in the subset of the data used. $z = 0$ corresponds to $100$ samples. Note that for the finite fidelity methods MF-SKO and MF-GP-UCB, approximations are obtained at $z = 0.33$ and $z = 0.667$.

For our algorithms we set $\nu_{max} = 1.0$ and $\rho_{max} = 0.9$. At the beginning of the experiment some of the budget is expended to obtain the function values at a point $x$ with two different fidelities $z_1 = 0.8$ and $z_2 = 0.2$. Thus the total budget spent in the initialization is $\Lambda(0.8) + \lambda(0.2)$. The function values obtained are then used to initialize $c$ in the bias function $\zeta(z) = c(1 - z)$. The initial value of $c$ is set to $2|f_1 - f_2|/|z_1 - z_2|$. Thereafter, $c$ is updated online according to the method detailed above. We set $N = 0.5D_{max}\log(\Lambda/\lambda(1))$.

The cross-validation accuracy obtained as a function of time is plotted in Fig. 1f for all the candidate algorithms. It can be observed that MFPDOO outperforms the other algorithms, especially in low-budget settings.

## 7. Conclusion

We considered the problem of black-box function optimization using hierarchical partitions in the presence of cheap approximations or fidelities. We propose two *tree-search* based algorithms which can navigate the domain effectively using cheaper fidelities for coarser partitions and more expensive ones while zeroing in on finer partitions. We analyze our algorithms under standard smoothness assumptions and provide simple regret guarantees given a cost budget $\Lambda$. Our simple regret guarantees scale much better with respect to $\Lambda$ as compared to other hierarchical partitioning based algorithms that do not use cheaper fidelities. Our first algorithm (MFDOO) requires the knowledge of the smoothness parameters $(\nu_*, \rho_*)$ and has a simple regret bound of $\mathcal{O}(\Lambda^{-1/(d(\nu_*,\rho_*)+1)})$ where $d(\nu_*, \rho_*)$ is the near-optimality dimension. Our second algorithm (MFPDOO) can obtain a simple regret bound of $\mathcal{O}((\Lambda/\log\Lambda)^{-1/(d(\nu_*,\rho_*)+1)})$ even when the smoothness parameter are unknown. Finally, we empirically validate the performance of our algorithms on real and synthetic datasets, where they outperform the state-of-the art multi-fidelity algorithms.

This work opens up several interesting research directions. The theoretical guarantees of our algorithms assume some nice properties about the bias and cost functions. We believe it is possible to design more robust algorithms that have similar guarantees even for the bias and cost functions that are not well-designed. Our setting is also restricted to a one dimensional fidelity space. However, in many application the fidelity space may be multi-dimensional. For instance, in the hyper-parameter tuning one can choose to use less samples or train for lesser iterations. It is an interesting research direction to incorporate a multi-dimensional fidelity space with tree-search based algorithms. Finally, in this work we work in the noise-less setting where the function and the approximations are deterministic. We believe it is possible to extend our results to a setting where zero-mean noise is added to the function and its approximations.

**Acknowledgment**

This work is partially supported by NSF grant 1320175, ARO grant W911NF-17-1-0359, and the US DoT supported D-STOP Tier 1 University Transportation Center.

## References

Agarwal, Alekh, Duchi, John C, Bartlett, Peter L, and Levrard, Clement. Oracle inequalities for computationally budgeted model selection. In *COLT*, 2011.

Bubeck, Sébastien, Munos, Rémi, Stoltz, Gilles, and Szepesvári, Csaba. X-armed bandits. *Journal of Machine Learning Research*, 12(May):1655–1695, 2011.

Currin, Carla. A bayesian approach to the design and analysis of computer experiments. Technical report, ORNL Oak Ridge National Laboratory (US), 1988.

Cutler, Mark, Walsh, Thomas J., and How, Jonathan P. Reinforcement Learning with Multi-Fidelity Simulators. In *ICRA*, 2014.

Dixon, L. C. W. and Szego, George Philip. *Towards global optimisation 2*, volume 2. North Holland, 1978.

Finkel, Daniel E. Direct optimization algorithm user guide. *Center for Research in Scientific Computation, North Carolina State University*, 2, 2003.

Forrester, Alexander I. J., Sóbester, András, and Keane, Andy J. Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, 2007.

Grill, Jean-Bastien, Valko, Michal, and Munos, Rémi. Black-box optimization of noisy functions with unknown smoothness. In *Advances in Neural Information Processing Systems*, pp. 667–675, 2015.

Huang, D., Allen, T.T., Notz, W.I., and Miller, R.A. Sequential kriging optimization using multiple-fidelity evaluations. *Structural and Multidisciplinary Optimization*, 2006a.

Huang, Deng, Allen, Theodore T, Notz, William I, and Miller, R Allen. Sequential kriging optimization using multiple-fidelity evaluations. *Structural and Multidisciplinary Optimization*, 32(5):369–382, 2006b.

Joachims, Thorsten. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, Carnegie-mellon univ pittsburgh pa dept of computer science, 1996.

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

Kandasamy, Kirthevasan, Dasarathy, Gautam, Oliva, Junier B, Schneider, Jeff, and Póczos, Barnabás. Gaussian process bandit optimisation with multi-fidelity evaluations. In *Advances in Neural Information Processing Systems*, pp. 992–1000, 2016a.

Kandasamy, Kirthevasan, Dasarathy, Gautam, Oliva, Junier B, Schneider, Jeff, and Poczos, Barnabas. Multi-fidelity gaussian process bandit optimisation. *arXiv preprint arXiv:1603.06288*, 2016b.

Kandasamy, Kirthevasan, Dasarathy, Gautam, Poczos, Barnabas, and Schneider, Jeff. The multi-fidelity multi-armed bandit. In *Advances in Neural Information Processing Systems*, pp. 1777–1785, 2016c.

Kandasamy, Kirthevasan, Dasarathy, Gautam, Schneider, Jeff, and Poczos, Barnabas. Multi-fidelity bayesian optimisation with continuous approximations. *arXiv preprint arXiv:1703.06240*, 2017.

Klein, Aaron, Falkner, Stefan, Bartels, Simon, Hennig, Philipp, and Hutter, Frank. Fast bayesian optimization of machine learning hyperparameters on large datasets. *arXiv preprint arXiv:1605.07079*, 2016.

Kleinberg, Robert, Slivkins, Aleksandrs, and Upfal, Eli. Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pp. 681–690. ACM, 2008.

Kocsis, Levente and Szepesvári, Csaba. Bandit based monte-carlo planning. In *European conference on machine learning*, pp. 282–293. Springer, 2006.

Lam, Rémi, Allaire, Douglas L, and Willcox, Karen E. Multifidelity optimization using statistical surrogate modeling for non-hierarchical information sources. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, pp. 0143, 2015.

Li, Lisha, Jamieson, Kevin, DeSalvo, Giulia, Rostamizadeh, Afshin, and Talwalkar, Ameet. Hyperband: A novel bandit-based approach to hyperparameter optimization. *arXiv preprint arXiv:1603.06560*, 2016.

Martinez-Cantin, R., de Freitas, N., Doucet, A., and Castellanos, J. Active Policy Learning for Robot Planning and Exploration under Uncertainty. In *Proceedings of Robotics: Science and Systems*, 2007.

Munos, Rémi. Optimistic optimization of a deterministic function without the knowledge of its smoothness. In *Advances in neural information processing systems*, pp. 783–791, 2011.

Parkinson, David, Mukherjee, Pia, and Liddle, Andrew R. A Bayesian model selection analysis of WMAP3. *Physical Review*, 2006.

Poloczek, Matthias, Wang, Jialei, and Frazier, Peter I. Multi-information source optimization. *arXiv preprint arXiv:1603.00389*, 2016.

Sabharwal, A, Samulowitz, H, and Tesauro, G. Selecting near-optimal learners via incremental data allocation. In *AAAI*, 2015.

Slivkins, Aleksandrs. Multi-armed bandits on implicit metric spaces. In *Advances in Neural Information Processing Systems*, pp. 1602–1610, 2011.

Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems*, 2012.

Srinivas, Niranjan, Krause, Andreas, Kakade, Sham M, and Seeger, Matthias. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Valko, Michal, Carpentier, Alexandra, and Munos, Rémi. Stochastic simultaneous optimistic optimization. In *International Conference on Machine Learning*, pp. 19–27, 2013.

Zhang, C. and Chaudhuri, K. Active Learning from Weak and Strong Labelers. In *NIPS*, 2015.