Procrastinated Tree Search: Black-box Optimization with Delayed, Noisy, and Multi-fidelity Feedback

Junxiong Wang
Dept. of Computer Science
Cornell University
Ithaca, NY, USA 14850

Debabrota Basu
Équipe Scool, Inria
UMR 9189 - CRIStAL, CNRS
Univ. Lille, Centrale Lille
Lille, France 59000

Immanuel Trummer
Dept. of Computer Science
Cornell University
Ithaca, NY, USA 14850

Abstract

In black-box optimization problems, we aim to maximize an unknown objective function, where the function is only accessible through feedbacks of an evaluation or simulation oracle. In real-life, the feedbacks of such oracles are often noisy and available after some unknown delay that may depend on the computation time of the oracle. Additionally, if the exact evaluations are expensive but coarse approximations are available at a lower cost, the feedbacks can have multi-fidelity. In order to address this problem, we propose a generic extension of hierarchical optimistic tree search (HOO), called ProCrastinated Tree Search (PCTS), that flexibly accommodates a delay and noise-tolerant bandit algorithm. We provide a generic proof technique to quantify regret of PCTS under delayed, noisy, and multi-fidelity feedbacks. Specifically, we derive regret bounds of PCTS enabled with delayed-UCB1 (DUCB1) and delayed-UCB-V (DUCBV) algorithms. Given a horizon T, PCTS retains the regret bound of non-delayed HOO for expected delay of $O(\log T)$ and worsens by $O(T^{\frac{1-\alpha}{d+2}})$ for expected delays of $O(T^{1-\alpha})$ for $\alpha \in (0, 1]$. We experimentally validate on multiple synthetic functions and hyperparameter tuning problems that PCTS outperforms the state-of-the-art black-box optimization methods for feedbacks with different noise levels, delays, and fidelity.

1 Introduction

Black-box optimization (Munos, 2014; Sen et al., 2019), alternatively known as zeroth-order optimization (Xu et al., 2020) or continuous-arm multi-armed bandit (Bubeck et al., 2011), is a widely studied problem and has been successfully applied in reinforcement learning (Munos, 2014; Grill et al., 2020), neural architecture search (Wang et al., 2019a), large-scale database tuning (Pavlo et al., 2017; Wang et al., 2021), robotics (Martinez-Cantin, 2017), AutoML (Fischer et al., 2015), material science (Xue et al., 2016; Kajita et al., 2020), and many other domains. In black-box optimization, we aim to maximize an unknown function $f : \mathcal{X} \to \mathbb{R}$, i.e. to find

$$x^* \triangleq \arg\max_{x \in \mathcal{X}} f(x). \tag{1}$$

In this setting, the optimizer does not have access to the derivatives of f, rather can access f only by sequentially querying a simulation or evaluation oracle (Jamieson et al., 2012). The goal is to minimize the expected error in optimization, i.e. $\mathbb{E}[f(x^*) - f(x_T)]$, after T queries (Munos, 2014), or to reach a fixed error threshold with as less number of queries a possible (Jamieson et al., 2012). We adopt the first approach of analysis in this paper.

Approaches to Black-box Optimization. Jamieson et al. (2012) has shown that doing black-box optimization for convex functions is in general efficient. For convex functions, typically Zeroth-

Preprint. Under review.

Table 1: Comparison of existing tree search, BO, and zeroth-order GD optimizers.

Algorithm	Expected Simple Regret	Delay	Noise	Fidelity	Assumptions
PCTS	$T^{-1/(d+2)}(\log T + \frac{\mathbb{E}[Delay]}{\sigma^2+2b})^{1/(d+2)}$	Stochastic	Unknown	Yes	Local Lip.
HOO (Bubeck et al., 2011)	$T^{-1/(d+2)}(\log T)^{1/(d+2)}$	x	Known	MF-HOO (Sen et al., 2019)	Local Lip.
GP-UCB (Srinivas et al., 2010)	$T^{-1/2}\,\mathrm{InfoGain}(T)$	x	Known	MF-GP-UCB (Kandasamy et al., 2016)	GP surrogate
GP-EI (Jones et al., 1998; Nguyen et al., 2019)	$T^{-1/2}O((\log T)^{d/2})$	x	Known	x	GP surrogate
DBGD (Li et al., 2019)	$\sqrt{T+\sum Delay}/T$	Bounded	x	x	Convex

order (ZO) Gradient Descent (GD) framework is used that replaces the gradient with the difference between functional evaluations (Jamieson et al., 2012; Kumagai, 2017). This approach requires double evaluation queries per-step and also multiple problem-specific hyperparameters to be tuned to obtain reasonable performance. Still, these methods are less robust to noise and stochastic delay (Li et al., 2019) than the next two other approaches, i.e. Bayesian Optimization (BO) and Hierarchical Tree Search.

For an objective function with no known structure except local smoothness, solving the black-box optimization problem is equivalent to estimating f almost everywhere in its domain $\mathcal{X}$ (Goldstein, 1977). This can lead to an exponential complexity in the dimensionality of the domain (Chen, 1988; Wang et al., 2019b). Thus, one approach for this problem is to learn a surrogate $\hat{f}$ of the actual function f, such that $\hat{f}$ is a close approximation of f and $\hat{f}$ can be learned and optimized with fewer samples. This has led to research in Bayesian Optimization (BO) and its variants (Srinivas et al., 2010; Jones et al., 1998; Huang et al., 2006; Kandasamy et al., 2016), where specific surrogate regressors are fitted to the Bayesian posterior of f. However, if f is highly nonlinear or high dimensional, the Bayesian surrogate, namely Gaussian Process (GP) (Srinivas et al., 2010) or Bayesian Neural Network (BNN) (Springenberg et al., 2016), requires many samples to fit and generalize well. Also, there are two other issues. Firstly, often myopic acquisition in BO algorithms to explore the boundary of the search domain excessively (Oh et al., 2018). Secondly, the error bounds of BO algorithms include the information gain term ($\mathrm{InfoGain}(T)$) that often increases with T (Srinivas et al., 2010).

Instead of fixing on to such specific surrogate modelling, the alternative is to use *hierarchical tree search* methods which have drawn significant attention and success in recent past (Munos, 2014; Bubeck et al., 2011; Kleinberg et al., 2008a; Grill et al., 2015; Shang et al., 2018, 2019; Sen et al., 2018, 2019).The tree search approach explores the space using a hierarchical binary tree with nodes representing subdomains of the function domain $\mathcal{X}$. Then, it leverages a bandit algorithm to balance the exploration of the domain and fast convergence towards the subdomains with optimal values of f. This approach does not demand more than local smoothness assumption with respect to the hierarchical partition (Shang et al., 2018, Assumption 1) and an asymptotically consistent bandit algorithm (Bubeck et al., 2011). The generic nature of hierarchical tree search motivated us to extend it to black-box optimization with delayed, noisy, and multi-fidelity feedbacks.

Imperfect Oracle: Delay, Noise, and Multi-fidelity (DNF). In real-life, the feedbacks of the evaluation oracle can be received after a delay due to the computation time to complete the simulation or evaluation (Weinberger and Ordentlich, 2002), or to complete the communication between servers (Agarwal and Duchi, 2012; Sra et al., 2015). Such delayed feedback is natural in different optimization problems, including the white-box settings (Wang et al., 2021; Li et al., 2019; Joulani et al., 2016; Langford et al., 2009). In some other problems, introducing artificial delays while performing tree search, may create opportunities for work sharing between consecutive evaluations, thereby reducing computation time (Wang et al., 2021). This motivated us to look into the delayed feedback for black-box optimization. Additionally, the feedbacks of the oracle can be noisy or even the objective function itself can be noisy, for example simulation oracles for physical processes (Kajita et al., 2020) and evaluation oracles for hyperparameter tuning of classifiers (Sen et al., 2019) and computer systems (Fischer et al., 2015; Wang et al., 2021). On the other hand, the oracle may invoke a multi-fidelity framework. Specially, if there is a fixed computational or time budget for the optimization, the optimizer may choose to access coarse but cheaper evaluations of f than the exact and costlier evaluations (Sen et al., 2018, 2019; Kandasamy et al., 2016). Both the noisy functions and multi-fidelity frameworks are studied separately in tree search regime but mostly with an assumption of known upper bound on the noise variance (Sen et al., 2019) or known range of noise (Xu et al., 2020). We propose to extend tree search to a setting where all three imperfections, delay, noise, and multi-fidelity (*DNF*), are encountered concurrently. Additionally, we remove the requirement that the noise is either known or bounded.

Our Contributions. The main contributions of this paper are as follows:

1. *Algorithmic:* We show that the hierarchical tree search (HOO) framework is extendable to delayed, noisy and multi-fidelity (DNF) feedback through deployment of the upper confidence bounds of a bandit algorithm that is immune to the corresponding type of feedback. This reduces the tree search design problem to designing compatible bandit algorithms. In Section 3.1, we describe this generic framework, and refer to it as the *Procrastinated Tree Search* (PCTS).

2. *Theoretical:* We leverage the generalities of the regret analysis of tree search and incorporate delay and noise-tolerant bandit algorithms in it to show the expected simple regret bounds for expected delay $\tau = O(\log T)$ and $O(T^{1-\alpha})$ for $\alpha \in (0, 1)$. We instantiate the analysis for delayed versions of UCB1-σ (Auer et al., 2002) and UCB-V (Audibert et al., 2007). This requires analysing a delayed version of UCB1-σ and extending UCB-V to delayed setting. We show that we have constant loss and $T^{(1-\alpha)/(d+2)}$ loss compared to non-delayed HOO in case of the two delay models (Sec. 3.2). We also extend the analysis for unknown noise variance (Sec. 3.3) and multi-fidelity (Sec. 3.4). To the best of our knowledge, we are the first ones to consider DNF-feedback in hierarchical tree search, and our regret bound is more general than the existing ones for black-box optimization with either delay or known noise or multi-fidelity (Table 1).

3. *Experimental:* We experimentally and comparatively evaluate performance of PCTS on multiple synthetic and real-world hyperparameter optimization problems against the state-of-the-art black-box optimization algorithms (Sec. 4)[1]. We evaluate for different delays, noise variances (known and unknown), and fidelities. In all the experiments, PCTS with delayed-UCB1-σ (DUCB1σ) and delayed-UCB-V (DUCBV) outperform the competing tree search, BO, and zeroth-order GD optimizers.

2 Background and Problem Formulation

We aim to maximize an objective function $f : \mathcal{X} \leftarrow \mathbb{R}$, where the domain $\mathcal{X} \subseteq \mathbb{R}^D$. At each iteration, the algorithm queries f at a chosen point $x_t \in \mathcal{X}$ and gets back an evaluation $y = f(x_t)+\epsilon$, such that $\mathbb{E}[\epsilon] = 0$ and $\mathbb{V}[\epsilon] = \sigma^2$ (Jamieson et al., 2012). We consider both the cases, where σ^2 is known and unknown to the algorithm. We denote x^* as the optimum and $f^* \triangleq f(x^*)$ as the optimal value.

Structures of the Objective Function. In order to prove convergence of PCTS to the global optimal f^*, we need to assume that the domain $\mathcal{X}$ of f has at least a semi-metric ℓ defined on it (Munos, 2014). This allows us to define an ℓ-ball of radius ρ is $\mathcal{B}_\rho \triangleq \{x | \max_y \ell(x, y) \leq \rho \; \forall \; x, y \in \mathcal{B}_\rho \subseteq \mathcal{X}\}$. Now, we aim to define the near-optimality dimension of the function f, given semi-metric ℓ. The near-optimality dimension quantifies the inherent complexity of globally optimising a function using tree search type algorithms. Near-optimality dimension quantifies the ϵ-dependent growth in the number of ℓ-balls needed to pack this set of ϵ-optimal states: $\mathcal{X}_\epsilon \triangleq \{x \in \mathcal{X} | f(x) \geq f^* - \epsilon\}$.

Definition 1 (c-near-optimality dimension (Bubeck et al., 2011))**.** *c-near-optimality dimension is the smallest $d \geq 0$, such that for all $\epsilon > 0$, the maximal number of disjoint ℓ-balls of radius $c\epsilon$ whose centers can be accommodated in $\mathcal{X}_\epsilon$ is $O(\epsilon^{-d})$.*

This is a joint property of f and the dissimilarity measure ℓ. d is independent of the algorithm of choice and can be defined for any f and $\mathcal{X}$ with semi-metric ℓ. Additionally, we need f to be smooth around the optimum x^*, i.e. to be weak Lipschitz continuous, for the tree search to converge.

Assumption 1 (Weak Lipschitzness of f (Bubeck et al., 2011))**.** *For all $x, y \in \mathcal{X}$, f satisfies $f^* - f(y) \leq f^* - f(x) + \max\{f^* - f(x), \ell(x, y)\}$.*

Weak Lipschitzness implies that there is no sudden drop or jump in f around the optimum x^*. Weak Lipschitzness can hold even for discontinuous functions. Thus, it widens applicability of hierarchical tree search methodology and corresponding analysis to more general performance metrics and domain spaces in comparison with algorithms that explicitly need gradients or smoothness in stricter forms. In Appendix, we show that we can relax this assumption proposed in HOO to more local assumptions like (Shang et al., 2018). But here we keep this to directly compare the effects on HOO due to DNF feedbacks.

[1]Link to our code: `https://drive.google.com/drive/folders/188K6BoznXkdEWi8IOLclSNIktW_refR7`

Structure: Non-increasing Hierarchical Partition. The Hierarchical Tree Search (HOO) or $\mathcal{X}$-armed bandit family of algorithms (Bubeck et al., 2011; Shang et al., 2019; Sen et al., 2019) grow a tree $\mathcal{T} \subseteq \cup\{(h,l)\}_{h,l=0,1}^{H,2^h}$ of depth H, such that each node (h,l) represents a subdomain $\mathcal{X}_{(h,l)}$ of $\mathcal{X}$,[2] and the corresponding upper confidence intervals partition the domain of the performance metric f. Then, it uses a UCB-type bandit algorithm to assign optimistic upper confidence values to each partition. Using these values, it chooses a node to evaluate and expand at every time step. As the tree grows deeper, we obtain a more granular hierarchical partition of the domain. As we want the confidence intervals to shrink with increase in their depth, we need to ensure certain regularity of such hierarchical partition. Though we state the hierarchical partition as an assumption, it can be considered as an artifact of the tree search algorithm.

Assumption 2 (Hierarchical Partition with Decreasing Diameter and Shape (Munos, 2014))**.**

1. Decreasing diameters. *There exists a decreasing sequence $\delta(h) > 0$ and constant $\nu_1 > 0$ such that* $\text{diam}(X_{h,l}) \triangleq \max_{x \in X_{h,l}} \ell(x_{h,l}, x) \leq \nu_1 \delta(h)$*,for any depth $h \geq 0$, for any interval $X_{h,l}$, and for all $i = 1, \ldots, 2^h$. For simplicity, we consider that $\delta(h) = \rho^h$ for $\rho \in (0,1)$.*

2. Regularity of the intervals. *There exists a constant $\nu_2 > 0$ such that for any depth $h \geq 0$, every interval $X_{h,l}$ contains at least a ball $\mathcal{B}_{h,l}$ of radius $\nu_2 \rho^h$ and center $x_{h,l}$ in it. Since the tree creates a partition at any given depth h, $\mathcal{B}_{h,l} \cap \mathcal{B}_{h,l'} = \emptyset$ for all $1 \leq l < l' \leq 2^h$.*

Simple Regret: Performance Metric. While analyzing iterative or sequential algorithms, regret $\text{Reg}_T \triangleq \sum_{t=1}^{T} [f(x^*) - f(x_t)]$ is widely used as the performance measure (Munos, 2014). For optimization algorithms, another relevant performance metric is expected error or expected simple regret incurred at time T: $\epsilon_T = \mathbb{E}[r_T] = \mathbb{E}[f(x^*) - f(x_T)] = \frac{1}{T}\mathbb{E}[\text{Reg}_T]$. Since the last equality holds for tree search (Munos, 2014), we state only the expected simple regret results in the main paper. The algorithm performance is better if the expected simple regret is lower. If the upper bound on expected simple regret grows sublinearly with horizon T, the corresponding algorithm asymptotically converges to the optimum. Given the aforementioned assumptions and definitions, and choosing simple regret as the performance measure, we state the expected error bound of HOO (Bubeck et al., 2011, Thm. 6) (using UCB1).

Theorem 1 (Regret of HOO)**.** *Assume that the expected objective function f satisfies Assumption 1, and its $4\nu_1/\nu_2$-near-optimality dimension is $d > 0$. Then, under Assumption 2 and for any $d' > d$, expected simple regret of HOO*

$$\epsilon_T = \mathbb{E}[r_T] = O\left(T^{-\frac{1}{d'+2}} (\log T)^{\frac{1}{d'+2}}\right) \tag{2}$$

for a horizon $T > 1$, and $4\nu_1/\nu_2$-near-optimality dimension d of f.

3 PCTS: Procrastinated Tree Search

In this section, we first provide a generic template for our framework. Following that, we incrementally show expected error bounds under delayed, noisy with known variance, noisy with unknown variance, and multi-fidelity feedbacks.

3.1 Algorithmic Framework

PCTS adapts the HOO algorithm (Bubeck et al., 2011) to delayed, noisy, and multi-fidelity feedbacks. We illustrate the pseudocode in Algorithm 1. Thus, in PCTS, we first assign optimistic $B^{\min}$ values to each node (h,l) in the hierarchical tree $\mathcal{T}_t$. Then, we incrementally select an 'optimistic path' from the root such that the path corresponds to one node at every depth and every chosen node has larger $B^{\min}$ value than its sibling nodes. Sibling nodes are the nodes that share the same parent. Following that, we sample a point x_t randomly from the subdomain $X_{(h_t,l_t)}$ that the leaf node (h_t, l_t) of the optimistic path represents. We expand this leaf node and add its children to $\mathcal{T}_{t+1}$. Lines 4-6 and 9 essentially comes from the HOO algorithm. The difference is in mainly three steps. In Line 7, we query the evaluation oracle with the point x_t and fidelity z_{h_t} due to multi-fidelity evaluator. In Line 8, we observe a delayed set of noisy feedbacks $\mathcal{O}_t \triangleq \{f_{s|t}(x_{h_s,l_s}|z_{h_s}) + \epsilon_s : s + \tau_s = t\}$ that arrives with corresponding timestamps when the queries were invoked. Here, $f_{s|t}(x_{h_s,l_s}|z_{h_s})$ is the

[2]Here, (h,l) represents the l-th node at depth h.

Algorithm 1 PCTS under DNF feedback and with a compatible BANDIT algorithm

1: **Input:** Total cost budget Λ, Bias function ζ, Cost function λ, Smoothness parameters (ν_1, ρ).
2: **Initialization:** $\mathcal{T}_1 = \{(0,0)\}$ (root), $B^{\min}_{(1,1)} = B^{\min}_{(1,2)} = \infty$, $t = 0$ (iteration), $C = 0$ (cost)
3: **while** $C \leq \Lambda$ **do**
4: Compute $B^{\min}$ values for each node in $\mathcal{T}_t$ using a UCB-type algorithm BANDIT(Eq. (3))
5: Select a leaf node (h_t, l_t) by following an "optimistic path" from root such that each selected node in the path has the highest $B^{\min}$ value among its sibling nodes
6: Sample a point x_t uniformly at random in the subdomain of node (h_t, l_t)
7: Query the evaluation oracle with x_t and at fidelity z_{h_t}
8: Observes the delayed and noisy feedbacks $\mathcal{O}_t \triangleq \{f_{s|t}(x_{h_s,l_s}|z_{h_s}) + \epsilon_s : s + \tau_s = t\}$ with the timestamps of invoking these queries $\{s : s + \tau_s = t\}$.
9: Expand node (h_t, l_t) and add its children to $\mathcal{T}_t$ to form $\mathcal{T}_{t+1}$
10: **end while**

multi-fidelity feedback lower bounded by $f(x_{h_s,l_s}) - \zeta(z_{h_s})$, and ϵ_s is a noise with zero mean and bounded variance. Such DNF feedback constrains us to use an asymptotically optimal bandit algorithm, BANDIT (Line 4), that allows us to get an upper confidence bound $B_{(h,l),s,t}$, which would be immune to DNF. In the following sections, we incrementally design such BANDIT confidence bounds and derive corresponding error bounds for PCTS. Though we describe the algorithm and the analysis for given smoothness parameters (ν_1, ρ), we describe in Appendix B the details of how to extend PCTS to unknown smoothness parameters.

3.2 Adapting to Delayed Feedbacks

Observable Stochastic Delay Model. We consider the stochastic delay setting (Joulani et al., 2013, 2016). This means that the feedback $f(x_s)$ of the evaluation oracle invoked at time $s \in [0, T]$ arrives with a delay $\tau_s \in \mathbb{R}^{\geq 0}$, such that $\{\tau_s\}_{s=0}^{T}$ are random variables invoked by an underlying but unknown stochastic process $\mathcal{D}$. Here, the delays are independent of the algorithm's actions.

Assumption 3 (Bounded Mean Delay)**.** *Delays are generated i.i.d from an unknown delay distribution $\mathcal{D}$. The expectation of delays $\tau \triangleq \mathbb{E}[\tau_s : s \geq 0]$ is bounded and observable to the algorithm.*

We observe that constant or deterministic delay with $\tau_{const} < \infty$ is a special case of this delay model.

From PCTS to Delayed Bandits. Due to the delayed setting, we observe feedback of a query invoked at time s at time $t \geq s$. Let us denote such feedback as $f_{s|t}(x_s)$. Thus, at time t, PCTS does not have access to all the invoked queries but a delayed subset of it: $\cup_{t=1}^{t} \mathcal{O}_{t'} = \cup_{t=1}^{t} \{f_{s|t'}(x_s) : s + \tau_s = t'\}$. At time t, PCTS uses $\mathcal{O}_t$ to decide which node to extend next. Thus, making PCTS immune to unknown stochastic delays reduces to deployment of a BANDIT algorithm that can handle such stochastic delayed feedback.

Multi-armed bandits with delayed feedback is an active research area (Eick, 1988; Joulani et al., 2013; Vernade et al., 2017; Gael et al., 2020; Pike-Burke et al., 2018), where researchers have incrementally studied the known constant delay, the unknown observable stochastic delay, and the unknown anonymous stochastic delay settings. In this paper, we operate in the second setting, where a delayed feedback comes with the timestamp of the query. Under delayed feedback, designing an UCB-type bandit algorithm requires defining an optimistic confidence interval around the expected value of a given node i that will consider both $T_i(t)$ and $S_i(t)$. $T_i(t)$ and $S_i(t)$ are the number of times a node i is evaluated and the number of evaluation feedbacks observed until time t.

Given such delayed statistics, any UCB-like bandit algorithm computes $B_{i,s,t}$, i.e. the optimistic upper confidence bound for action i at time t (Table 2), and chooses the one with maximum $B_{i,s,t}$:

$$i_t = \arg\max_{i \in \mathcal{A}} B_{i,s,t}.$$

We show three such confidence bounds in Table 2. Here, $\hat{\mu}_{i,s}$, $\hat{\sigma}^2_{i,s}$, and σ^2 are sample mean, sample variance, and predefined variance respectively. For non-delayed setting, $s = T_i(t-1)$, and for delayed setting, $s = S_i(t-1)$. Representing the modifications of UCB1 (Auer et al., 2002),

Table 2: Confidence Bounds for different bandit algorithms with delayed/non-delayed feedback.

BANDIT	DUCB1 (Joulani et al., 2016)	DUCB1σ	DUCBV
$B_{i,s,t}$	$\hat{\mu}_{i,s} + \sqrt{\frac{2\log t}{s}}$	$\hat{\mu}_{i,s} + \sqrt{\frac{2\sigma^2 \log t}{s}}$	$\hat{\mu}_{i,s} + \sqrt{\frac{2\hat{\sigma}^2_{i,s} \log t}{s}} + \frac{3b \log t}{s}$

UCB1-σ (Auer et al., 2002), and UCB-V (Audibert et al., 2007) in such a general form allows us to extend them for delayed settings and incorporate them for node selection in PCTS. Thus, given an aforementioned UCB-like optimistic bandit algorithm, the leaf node (h_t, l_t) selected by PCTS at time t is

$$\begin{aligned}(h_t, l_t) &\triangleq \arg\max_{(h,l)\in\mathcal{T}_t} B^{\min}_{(h,l)}(t) \\ &\triangleq \arg\max_{(h,l)\in\mathcal{T}_t} \min\{B_{(h,l),s,t} + \nu_1 \rho^h, \max_{(h',l')\in C(h,l)} B^{\min}_{(h,l)}(t)\}. \end{aligned} \tag{3}$$

Here, $\mathcal{T}_t$ is the tree constructed at time t, and $C(h,l)$ is the set of children nodes of the node (h,l). Equation (3) is as same as that of HOO except that $B_{(h,l),s,t}$ is replaced by bounds in Table 2. Under these modified confidence bounds for delays, we derive the bound on expected regret of PCTS + DUCB1 that extends the regret analysis of bandits with delayed feedback to HOO (Munos, 2014).

Theorem 2 (Regret of PCTS + DUCB1 under Stochastic Delays)**.** *Under the same assumptions as Theorem 1 and upper bound on expected delay τ,* PCTS *using Delayed-UCB1 (*DUCB1*) achieves expected simple regret*

$$\epsilon_T = O\left(\left(\frac{\ln T}{T}\right)^{\frac{1}{d'+2}} \left(1 + \frac{\tau}{\ln T}\right)^{\frac{1}{d'+2}}\right). \tag{4}$$

Corollary 1 (Regret of PCTS + DUCB1 under Constant Delay)**.** *If the assumptions of Theorem 1 hold, and the delay is constant, i.e. $\tau_{const} > 0$, the expected simple regret of* PCTS + DUCB1 *is*

$$\epsilon_T = O\left(\left(\frac{\ln T}{T}\right)^{\frac{1}{d'+2}} \left(1 + \frac{\tau_{const}}{\ln T}\right)^{\frac{1}{d+2}}\right). \tag{5}$$

Consequences of Theorem 2. The bound of Theorem 2 provides us with a few interesting insights. 1. *Degradation due to delay:* we observe that the expected error of PCTS + DUCB1 worsens by a factor of $\left(1 + \frac{\tau}{\ln T}\right)^{\frac{1}{d+2}}$ compared to HOO, which uses the non-delayed UCB1 (Auer et al., 2002). This is still significantly better than the other global optimization algorithm that can handle delay, such as Delayed Bandit Gradient Descent (DBGD) (Li et al., 2019). As DBGD achieves expected error bound $\sqrt{\frac{1}{T} + \frac{D}{T}}$, where D is the total delay. Also, appearance of delay as an additive term in our analysis resonates with the proven results in bandits with delayed feedback,where an additive term appears in regret bounds due to delay. For $d = 0$, our bound matches in terms of T and τ with the problem-independent lower bound of bandits with finite K-arms and constant delay, i.e. $\sqrt{(K/T + \tau/T)}$ (Cesa-Bianchi et al., 2016, Cor. 11), up to logarithmic factors.

2. *Wait-and-act vs.* PCTS + DUCB1. A naïve way to handle *known constant delay* is to wait for the next τ_{const} time steps and to collect all the feedbacks in that interval to update the algorithm. In that case, the effective horizon becomes $\frac{T}{\tau_{const}}$. Thus, the corresponding error bound will be $O\left(T^{-\frac{1}{d'+2}} \left(\tau_{const} \ln T\right)^{\frac{1}{d+2}}\right)$. This is still higher than our error bound in Equation 5 for *unknown constant delay* $\tau_{const} > 1$ and $T \geq 3$.

3. *Deeper trees.* While proving Theorem 2, we observe that the depth $H > 0$ achieved by PCTS + DUCB1 till time T is such that $\rho^{-H(d'+2)} \geq \frac{T}{\tau + \ln T}$. This implies that for a fixed horizon T, the achieved depth should be $H \geq \frac{1}{d'+2} \frac{\tau + \ln T}{\ln(1/\rho)} = \Omega(\tau + \ln T)$. In contrast, HOO grows a tree of depth $H = \Omega(\ln(T/\tau))$. This shows that PCTS + DUCB1 constructs a deeper tree than HOO.

4. *Benign and adversarial delays.* If the expected delay is $O(\ln T)$, the expected simple regret is practically of the same order as that of non-delayed feedbacks. Thus, in cases of applications where introducing artificial delays helps in improving the computational cost (Wang et al., 2021), we can

Table 3: Per-step cost $\lambda(Z_h)$ and total number of iterations $H(\Lambda)$ for different Fidelity models.

Fidel. Model	Linear Growth	Constant	Polynomial Decay	Exponential Decay
$\lambda(Z_h)$	$\min\{\beta h, \lambda(1)\}, \beta > 0$	$\min\{\beta, \lambda(1)\}, \beta > 0$	$\min\{h^{-\beta}, \lambda(1)\}, \beta > 0, \neq 1$	$\min\{\beta^{-h}, \lambda(1)\}, \beta \in (0, 1]$
$H(\Lambda)$	$\sqrt{2(2\Lambda - \lambda(1))/\beta}$	$2(2\Lambda - \lambda(1))/\beta$	$(1 + (1-\beta)(2\Lambda - \lambda(1)))^{1/(1-\beta)}$	$\log_{1/\beta}(1 + (1-\beta)(2\Lambda - \lambda(1)))$

tune the delays to $O(\ln T)$ for a given horizon T without harming the accuracy. We refer to this range of delays as *benign delay*. In contrast, one can consider delay distributions that have tails with α-polynomial decay, i.e. the expected delay is $O(T^{1-\alpha})$ for $\alpha \in (0, 1)$. In that case, the expected error is at least $\tilde{O}(T^{-\frac{\alpha}{d+2}})$. Thus, it worsens the HOO bound by a factor of $T^{\frac{1-\alpha}{d+2}}$. This observation in error bound resonates with the impossibility result of (Gael et al., 2020) that, in case of delays with α-polynomial tails, a delayed bandit algorithm cannot achieve total expected regret lower than $(T^{1-\alpha})$. Thus, it is unexpected that any hierarchical tree search with such delays achieves expected error better than $O(T^{-\frac{\alpha}{d+2}})$.

3.3 Adapting to Delayed and Noisy Feedback

Typically, when we evaluate the objective function at any time step t, we obtain a noisy version of the function as feedback such that $\tilde{f}(X_t) = f(X_t) + \epsilon_t$. Here, ϵ_t is a noise sample independently generated from a noise distribution $\mathcal{N}$ with mean 0. Till now, we did not explicitly consider the noise for the action selection step. In this section, we provide analysis for both known and unknown variance cases. In both cases, we assume that the noise has bounded variance σ^2, i.e. sub-Gaussian. In general, this assumption can be imposed in the present setup as any noisy evaluation can be clipped in the range of the evaluations where we optimize the objective function. It is known that a bounded random variable is sub-Gaussian with bounded mean and variance.

Case 1: Known Variance. Let us assume that the variance of the noise is known, say $\sigma^2 > 0$. In this case, the optimistic B-values can be computed using a simple variant of delayed-UCB1, i.e. delayed-UCB-σ (DUCB1σ), where

$$B_{(h,i),S_{(h,i)}(t-1),t} \triangleq \hat{\mu}_{(h,i),S_{(h,i)}(t-1)} + \sqrt{\frac{2\sigma^2 \log t}{S_{(h,i)}(t-1)}}. \tag{6}$$

Here, $\hat{\mu}_{(h,i),S_{(h,i)}(t-1)}$ is the empirical mean computed using noisy evaluations obtained till time t, i.e. $\cup_{t'=0}^{t} \mathcal{O}_t$, and for node (h, i). In multiple works, this known noise setup and $UCB - \sigma^2$ algorithm has been considered in tree search algorithms without delays.

Theorem 3. *Let us assume that the variance of the noise in evaluations is σ^2 and is available to the algorithm. Then, under the same assumptions as Theorem 1 and upper bound on expected delay τ,* PCTS *using* DUCB1σ *for node selection achieves expected simple regret*

$$\epsilon_T = O\left(T^{-\frac{1}{d'+2}} \left((\sigma/\nu_1)^2 \ln T + \tau\right)^{\frac{1}{d+2}}\right). \tag{7}$$

Effect of Known Noise. We observe that even with no delay, i.e. $\tau = 0$, noisy feedback with known variance σ^2 worsens the bound of HOO with noiseless evaluations by $\sigma^{2/(d+2)}$.

Case 2: Unknown Variance. If the variance of the noise is unknown, we have to estimate the noise variance empirically from evaluations. Given the evaluations $\{\tilde{f}(X_1)\}_{t=0}^{T}$ and the delayed statistics $S_{(h,i)}(t-1)$ of node (h, i), the empirical noise variance at time t is $\hat{\sigma}^2_{(h,i),S_{(h,i)}(t-1)} \triangleq \frac{1}{S_{(h,i)}(t-1)} \sum_{j=1}^{S_{(h,i)}(t-1)} (\tilde{f}(X_j)\mathbb{1}[(H_j, I_j) = (h, i)] - \hat{\mu}_{(h,i),S_{(h,i)}(t-1)})^2$, where empirical mean $\hat{\mu}_{(h,i),S_{(h,i)}(t-1)} \triangleq \frac{1}{S_{(h,i)}(t-1)} \sum_{j=1}^{S_{(h,i)}(t-1)} \tilde{f}(X_j)\mathbb{1}[(H_j, I_j) = (h, i)]$. Using the empirical mean and variance of functional evaluations for each node (h, i), we now define a delayed-UCBV (DUCBV) confidence bound for selecting next node:

$$B_{(h,i),S_{(h,i)}(t-1),t} \triangleq \hat{\mu}_{(h,i),S_{(h,i)}(t-1)} + \sqrt{\frac{2\hat{\sigma}^2_{(h,i),S_{(h,i)}(t-1)} \log t}{S_{(h,i)}(t-1)}} + \frac{3b \log t}{S_{(h,i)}(t-1)}. \tag{8}$$

In practice, we do not need exact value of b. We can use a large proxy value such that the feedback is bounded by it.

Theorem 4. *Let us assume that the upper bound on variance of the noise in evaluations is σ^2, which is unknown to the algorithm. If $[0, b]$ is the range of f, under the same assumptions as of Theorem 1,* PCTS *using* DUCBV *achieves expected simple regret*

$$\epsilon_T = O\left(T^{-\frac{1}{d'+2}}\left(((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T + \tau\right)^{\frac{1}{d'+2}}\right).$$

Effect of Unknown Noise. *Adapting UCB-V in the stochastic delays and using the corresponding bound in* PCTS *allows us to extend hierarchical tree search for unknown noise both in presence and absence of delays.* In our knowledge, this paper is the first to extend HOO framework for unknown noise, and also UCB-V to stochastic delays. This adaptation to unknown noise comes at a cost of $(((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T)^{1/(d+2)}$ in expected error, whereas for known noise variance, it is $((\sigma/\nu_1)^2 \ln T)^{1/(d+2)}$.

3.4 Adapting to Delayed, Noisy, and Multi-fidelity (DNF) Feedback

Now, let us consider that we do not only have a delayed and noisy functional evaluator at each step but also an evaluator with different fidelity at each level h of the tree. This setup of multi-fidelity HOO without unknown noise and delay was first considered in (Sen et al., 2019). We extend their schematic to the version with delayed and noisy feedback with unknown delays and noise. Following the multi-fidelity formulation of (Sen et al., 2018, 2019), we consider the mean of the multi-fidelity query, $f_z(x)$, as biased, and progressively smaller bias can be obtained but with varying costs. The cost of selecting a new node at level $h > 0$ is $\lambda(Z_h) \in \mathbb{R}^+$ and the bias added in the decision due to the limited evaluation is $\zeta(Z_h) \in \mathbb{R}^+$. Here, the bias function is monotonically decreasing, and $Z_h \in \mathcal{Z}$ is the state of fidelity of the multi-fidelity evaluator, which influences both the cost of and the bias in evaluation. Thus, the evaluation at x_s is $\tilde{f}(x_{(h_s,l_s)}|z_{h,s}) \triangleq f(x_{(h_s,l_s)}) + \epsilon_s + \zeta(z_{h_s})$. Hence, under DNF feedback, the DUCBV selection rule becomes

$$B_{(h,i),S_{(h,i)}(t-1),t} \triangleq \hat{\mu}_{(h,i),S_{(h,i)}(t-1)} + \sqrt{\frac{2\hat{\sigma}^2_{(h,i),S_{(h,i)}(t-1)}\log t}{S_{(h,i)}(t-1)}} + \frac{3b\log t}{S_{(h,i)}(t-1)} + \zeta(Z_h). \quad (9)$$

Here, the empirical mean and variance are computed using the multi-fidelity and delayed feedbacks. We do not need to know ζ for the algorithm but we assume it to be known for the analysis. Given this update rule and the multi-fidelity model, we observe that the Lemma 1 of (Sen et al., 2019) holds. Given a total budget Λ and the multi-fidelity selection rule, the total number of iterations that the algorithms runs for is $T(\Lambda) \geq H(\Lambda) + 1$, where $H(\Lambda) \triangleq \max\{H : \sum_{h=1}^{H} \lambda(Z_h) \leq \Lambda\}$ Thus, we can retain the previously derived bounds of Theorem 2 and 4 by substituting $T = H(\Lambda)$.

Corollary 2 (PCTS + DUCBV under DNF Feedback)**.** *If the function under evaluation has h-dependent fidelity such that $H(\Lambda) \triangleq \max\{H : \sum_{h=1}^{H} \lambda(Z_h) \leq \Lambda\}$ and the induced bias $\zeta(Z_h) = \nu_1\rho^h$, then under the same assumptions as of Theorem 1,* PCTS *using* DUCB1 *achieves*

$$\epsilon_\Lambda = O\left((H(\Lambda))^{-\frac{1}{d'+2}}\left(\ln H(\Lambda) + \tau\right)^{\frac{1}{d'+2}}\right),$$

and PCTS *using* DUCBV *achieves expected simple regret*

$$\epsilon_\Lambda = O\left((H(\Lambda))^{-\frac{1}{d'+2}}\left(\ln H(\Lambda) + \frac{\tau}{(\sigma/\nu_1)^2 + 2b/\nu_1}\right)^{\frac{1}{d'+2}}\right).$$

Models of Multi-fidelity. Depending on the evaluation problem, we may have different cost functions. In Table 3, we instantiate the cost model, bias model, and total number of iterations for four multi-fidelity models with linear growth, constant, polynomially decaying, and exponentially decaying costs of evaluations. The linear growth, polynomial decays, and exponential decays are observed in the cases of hyperparameter tuning of deep-learning models, database optimization, and tuning learning rates of convex optimization respectively. Further details are in Appendix C.

4 Experimental Analysis

Experimental Setup. Similar to prior work on tree search with multi-fidelity and known noise (Sen et al., 2018, 2019), we evaluate performance of PCTS on both synthetic functions and machine

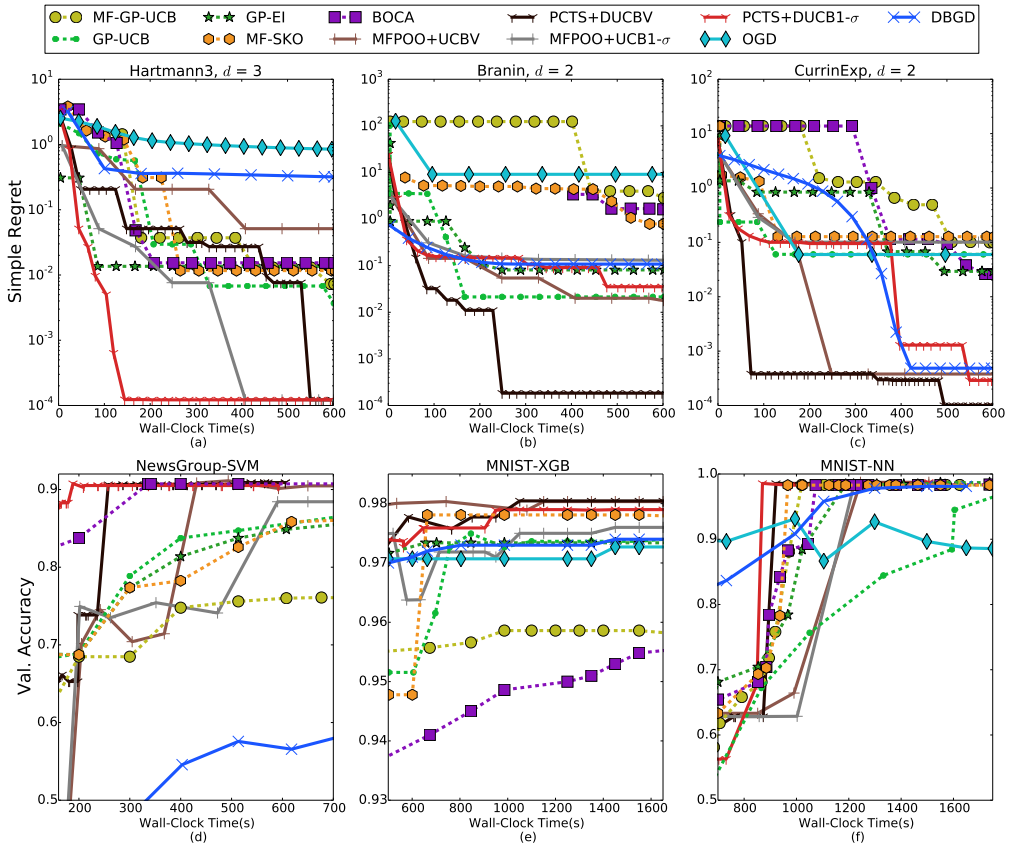

Figure 1: Figures (a) to (c) show simple regret (median of 10 runs) of different algorithms on synthetic functions with DNF feedbacks. Figures (d) to (f) show the cross-validation accuracy (median of 5 runs) achieved on the hyperparameter tuning of classifiers on datasets with DNF feedbacks.

learning models operating on real-data but under delayed, noisy (unknown), and multi-fidelity (DNF) feedback. We compare the performance of PCTS with: BO algorithms (BOCA (Kandasamy et al., 2017), GP-UCB (Srinivas et al., 2010), MF-GP-UCB (Kandasamy et al., 2016), GP-EI (Jones et al., 1998), MF-SKO (Huang et al., 2006))[3], tree search algorithms (MFPOO (Sen et al., 2018), MFPOO with UCB-V (Audibert et al., 2007)), zeroth-order GD algorithms (OGD, DBGD (Li et al., 2019)). In our experiments, we do not assume the smoothness parameters to be known. Rather, they are computed in a similar manner as POO and MFPOO. For comparison, we keep the delay constant and use wait-and-act versions of delay-insensitive baselines.

Synthetic Functions. We illustrate results for three different synthetic functions, Hartmann3 (van der Vlerk, 1996), Branin (van der Vlerk, 1996), and CurrinExp (Currin et al., 1988) with noise variances $\sigma^2 = 0.01$, 0.05, and 0.05 respectively. We follow the fidelity setup of (Sen et al., 2018, 2019), that modifies the synthetic functions to incorporate the fidelity space $\mathcal{Z} = [0, 1]$. The delay time τ for all synthetic functions is set to four seconds. We choose to add noise from Gaussian distributions with variance σ^2. Note that, the noise can be added from any distribution with variance $\leq \sigma^2$. This σ is passed to UCB1-σ and DUCB1-σ in MFPOO and PCTS as it assumes the noise variance is known (Sen et al., 2019). We implement all baselines in Python (version 2.7). We run each experiment ten times for 600s on a MacBook Pro with a 6-core Intel(R) Xeon(R)@2.60GHz CPU and plot the median value of simple regret, i.e. l_1 distance between the value of current best point and optimal value, for each algorithm.

Real Data: Hyperparameter Tuning. We evaluate the aforementioned algorithms on a 32-core Intel(R) Xeon(R)@2.3 GHz server for hyperparameter tuning of SVM on News Group dataset, and XGB and Neural Network on MNIST datasets. We use corresponding scikit-learn modules (Buitinck et al., 2013) for training all the classifiers. For each tuning task, we plot the median value of cross-

[3]We use the implementations in `https://github.com/rajatsen91/MFTreeSearchCV` for baselines except OGD, DBGD, and MFPOO-UCBV. For BO algorithms, this implementation chooses the best among the polynomial kernel, coordinate-wise product kernel and squared exponential kernel for each problem.

validation accuracy in five runs for 700s, 1700s, and 1800s respectively. We set $\sigma^2 = 0.02$ for algorithms where σ is known, and $b = 1$ where UCBV and DUCBVare used.

SVM on NewsGroup. We evaluate the algorithms to tune hyper-parameters of SVM classifier on the NewsGroup dataset (Lang, 1995). The hyper-parameters to tune are the regularization term C, ranging from $[e^{-5}, e^{5}]$, and the kernel temperature γ from the range $[e^{-5}, e^{5}]$. Both are accessed in log scale. We set the delay τ to four seconds and the fidelity range $\mathcal{Z} = [0, 1]$ is mapped to $[100, 7000]$. The fidelity range represents the number of samples used to train the SVM classifier with the chosen parameters. We plot the 5-fold cross-validation accuracy in Figure 1(d).

XGB on MNIST. We tune hyperparameters of XGBOOST (Chen and Guestrin, 2016) on the MNIST dataset (LeCun et al., 1998), where the hyperparameters are: (i) `max_depth` in $[2, 13]$, (ii) `n_estimators` in $[10, 400]$, (iii) `colsample_bytree` in $[0.2, 0.9]$, (iv) `gamma` in $[0, 0.7]$, and (v) `learning_rate` ranging from $[0.05, 0.3]$. The delay τ is set to ten seconds and the fidelity range $\mathcal{Z} = [0, 1]$ is mapped to the training sample range $[500, 20000]$. We plot the 3-fold cross-validation accuracy in Figure 1(e).

NN on MNIST. We also apply the algorithms for tuning the hyper-parameters of a three layer multi layer perceptron (MLP) neural network (NN) classifier on the MNIST dataset (LeCun et al., 1998). Here, the hyper-parameters being tuned are: (i) number of neurons of the first, second, and third layers, which belong to the ranges $[32, 128]$, $[128, 256]$, and $[256, 512]$, respectively, (ii) initial learning rate of optimizer in $[e^{-1}, e^{-5}]$ (accessed in log-scale), (iii) optimizers from ('lbfgs', 'sgd', 'adam'), (iv) activation function from ('tanh', 'relu', 'logistic'), (v) early_stopping from ('enable', 'disable'). The delay τ for this experiment is 20 seconds. The number of training samples corresponding to the fidelities $z = 0$ and 1 are 1000 and 60000 respectively. We plot the 3-fold cross-validation accuracy in Figure 1(f). **Summary of Results.** *In all of the experiments, we observe that either* PCTS + DUCB1 *or* PCTS + DUCBV *outperforms the competing algorithms in terms of convergence speed.* Also, in case of synthetic functions, they achieve approximately 1 to 3 order lower simple regret. These results empirically validate the efficiency of PCTS to adapt to DNF feedback. Due to lack of space, further implementation details, results on tree depth, performance for stochastic delays, and error statistics for other synthetic functions and hyperparamter tuning experiments are deferred to appendix.

5 Discussion and Future Work

We propose a generic tree search approach PCTS for black-box optimization problem with DNF feedback. We provide a generic analysis to bound the expected simple regret of PCTS given a horizon T. We instantiate PCTS with delayed-UCB1 and delayed-UCBV for observable stochastic delays, and known and unknown noises respectively. Our analysis shows that the expected simple regret for PCTS worsens by a constant factor and $T^{\frac{1-\alpha}{d+2}}$ for expected delay of $O(\log T)$ and $O(T^{1-\alpha})$ respectively. We also experimentally show that PCTS outperforms other global optimizers incompatible or individually tolerant to noise, delay, or multi-fidelity on both synthetic and real-world functions. In addition, our work extends UCB-V to stochastic delays.

In future, this work can be extended to anonymous delay feedbacks in order to develop tree search optimizers that respect privacy. It also shows need of proving a problem-independent lower bound for hierarchical tree search with stochastic delay. The other possible direction is to deploy PCTS for planning in Markov Decision Processes with delay, where tree search algorithms have been successful.

References

Agarwal, A. and Duchi, J. C. (2012). Distributed delayed stochastic optimization. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5451–5452. IEEE.

Agrawal, R. (1995). The continuum-armed bandit problem. *SIAM journal on control and optimization*, 33(6):1926–1951.

Audibert, J.-Y., Munos, R., and Szepesvári, C. (2007). Tuning bandit algorithms in stochastic environments. In Hutter, M., Servedio, R. A., and Takimoto, E., editors, *Algorithmic Learning Theory*, pages 150–165, Berlin, Heidelberg. Springer Berlin Heidelberg.

Auer, P., Cesa-bianchi, N., and Fischer, P. (2002). Finite time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256.

Auer, P., Ortner, R., and Szepesvári, C. (2007). Improved rates for the stochastic continuum-armed bandit problem. In *International Conference on Computational Learning Theory*, pages 454–468. Springer.

Azar, M. G., Lazaric, A., and Brunskill, E. (2014). Online stochastic optimization under correlated bandit feedback. In *International Conference on Machine Learning*, pages 1557–1565. PMLR.

Bubeck, S., Munos, R., Stoltz, G., and Szepesvári, C. (2011). X-armed bandits. *Journal of Machine Learning Research*, 12(5).

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., and Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122.

Cesa-Bianchi, N., Gentile, C., Mansour, Y., and Minora, A. (2016). Delay and cooperation in nonstochastic bandits. In *Conference on Learning Theory*, pages 605–622. PMLR.

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

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1988). A bayesian approach to the design and analysis of computer experiments. Technical report, Oak Ridge National Lab., TN (USA).

Eick, S. G. (1988). The two-armed bandit with delayed responses. *The Annals of Statistics*, pages 254–264.

Fischer, L., Gao, S., and Bernstein, A. (2015). Machines tuning machines: Configuring distributed stream processors with bayesian optimization. In *2015 IEEE International Conference on Cluster Computing*, pages 22–31. IEEE.

Gael, M. A., Vernade, C., Carpentier, A., and Valko, M. (2020). Stochastic bandits with arm-dependent delays. In *International Conference on Machine Learning*, pages 3348–3356. PMLR.

Goldstein, A. (1977). Optimization of lipschitz continuous functions. *Mathematical Programming*, 13(1):14–22.

Grill, J.-B., Altché, F., Tang, Y., Hubert, T., Valko, M., Antonoglou, I., and Munos, R. (2020). Monte-carlo tree search as regularized policy optimization. In *International Conference on Machine Learning*, pages 3769–3778. PMLR.

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

Hey, A. M. (1979). Towards global optimisation 2. *Journal of the Operational Research Society*, 30(9):844–844.

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

Jamieson, K. G., Nowak, R. D., and Recht, B. (2012). Query complexity of derivative-free optimization. *arXiv preprint arXiv:1209.2434*.

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *J. of Global Optimization*, 13(4):455492.

Joulani, P., Gyorgy, A., and Szepesvári, C. (2013). Online learning under delayed feedback. In *International Conference on Machine Learning*, pages 1453–1461. PMLR.

Joulani, P., Gyorgy, A., and Szepesvári, C. (2016). Delay-tolerant online convex optimization: Unified analysis and adaptive-gradient algorithms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Kajita, S., Kinjo, T., and Nishi, T. (2020). Autonomous molecular design by monte-carlo tree search and rapid evaluations using molecular dynamics simulations. *Communications Physics*, 3(1):1–11.

Kandasamy, K., Dasarathy, G., Oliva, J. B., Schneider, J., and Poczos, B. (2016). Gaussian process bandit optimisation with multi-fidelity evaluations. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Kandasamy, K., Dasarathy, G., Schneider, J., and Póczos, B. (2017). Multi-fidelity bayesian optimisation with continuous approximations. In *International Conference on Machine Learning*, pages 1799–1808. PMLR.

Kleinberg, R., Slivkins, A., and Upfal, E. (2008a). Multi-armed bandits in metric spaces. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, page 681690, New York, NY, USA. Association for Computing Machinery.

Kleinberg, R., Slivkins, A., and Upfal, E. (2008b). Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 681–690.

Kleinberg, R., Slivkins, A., and Upfal, E. (2019). Bandits and experts in metric spaces. *Journal of the ACM (JACM)*, 66(4):1–77.

Kumagai, W. (2017). Regret analysis for continuous dueling bandit. *arXiv preprint arXiv:1711.07693*.

Lang, K. (1995). Newsweeder: Learning to filter netnews. In *Machine Learning Proceedings 1995*, pages 331–339. Elsevier.

Langford, J., Smola, A., and Zinkevich, M. (2009). Slow learners are fast. *arXiv preprint arXiv:0911.0491*.

Lattimore, T. and Szepesvári, C. (2020). *Bandit algorithms*. Cambridge University Press.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Li, B., Chen, T., and Giannakis, G. B. (2019). Bandit online learning with unknown delays. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 993–1002. PMLR.

Locatelli, A. and Carpentier, A. (2018). Adaptivity to smoothness in x-armed bandits. In *Conference on Learning Theory*, pages 1463–1492. PMLR.

Martinez-Cantin, R. (2017). Bayesian optimization with adaptive kernels for robot control. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3350–3356. IEEE.

Munos, R. (2014). From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. *Foundations and Trends® in Machine Learning*, 7(1):1–129.

Nguyen, V., Gupta, S., Rana, S., Li, C., and Venkatesh, S. (2019). Filtering bayesian optimization approach in weakly specified search space. *Knowledge and Information Systems*, 60(1):385–413.

Oh, C., Gavves, E., and Welling, M. (2018). Bock: Bayesian optimization with cylindrical kernels. In *International Conference on Machine Learning*, pages 3868–3877. PMLR.

Pavlo, A., Angulo, G., Arulraj, J., Lin, H., Lin, J., Ma, L., Menon, P., Mowry, T. C., Perron, M., Quah, I., et al. (2017). Self-driving database management systems. In *CIDR*, volume 4, page 1.

Pike-Burke, C., Agrawal, S., Szepesvari, C., and Grunewalder, S. (2018). Bandits with delayed, aggregated anonymous feedback.

Sen, R., Kandasamy, K., and Shakkottai, S. (2018). Multi-fidelity black-box optimization with hierarchical partitions. In *International conference on machine learning*, pages 4538–4547. PMLR.

Sen, R., Kandasamy, K., and Shakkottai, S. (2019). Noisy blackbox optimization using multi-fidelity queries: A tree search approach. In *The 22nd international conference on artificial intelligence and statistics*, pages 2096–2105. PMLR.

Shang, X., Kaufmann, E., and Valko, M. (2018). Adaptive black-box optimization got easier: Hct only needs local smoothness. In *EWRL 2018*.

Shang, X., Kaufmann, E., and Valko, M. (2019). General parallel optimization without a metric. In *Algorithmic Learning Theory*, pages 762–788. PMLR.

Springenberg, J. T., Klein, A., Falkner, S., and Hutter, F. (2016). Bayesian optimization with robust bayesian neural networks. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Sra, S., Yu, A. W., Li, M., and Smola, A. J. (2015). Adadelay: Delay adaptive distributed stochastic convex optimization. *arXiv preprint arXiv:1508.05003*.

Srinivas, N., Krause, A., Kakade, S., and Seeger, M. (2010). Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 10151022, Madison, WI, USA. Omnipress.

van der Vlerk, M. H. (1996). Stochastic programming bibliography. *World Wide Web, http://mally. eco. rug. nl/spbib. html*, 2003.

Vernade, C., Cappé, O., and Perchet, V. (2017). Stochastic bandit models for delayed conversions. *arXiv preprint arXiv:1706.09186*.

Wang, J., Trummer, I., and Basu, D. (2021). UDO: universal database optimization using reinforcement learning. In *arXiv:2104.01744*, pages 1–13.

Wang, L., Zhao, Y., Jinnai, Y., Tian, Y., and Fonseca, R. (2019a). Alphax: exploring neural architectures with deep neural networks and monte carlo tree search. *arXiv preprint arXiv:1903.11059*.

Wang, Y., Balakrishnan, S., and Singh, A. (2019b). Optimization of smooth functions with noisy observations: Local minimax rates. *IEEE Transactions on Information Theory*, 65(11):7350–7366.

Weinberger, M. J. and Ordentlich, E. (2002). On delayed prediction of individual sequences. *IEEE Transactions on Information Theory*, 48(7):1959–1976.

Xu, Y., Joshi, A., Singh, A., and Dubrawski, A. (2020). Zeroth order non-convex optimization with dueling-choice bandits. In *Conference on Uncertainty in Artificial Intelligence*, pages 899–908. PMLR.

Xue, D., Balachandran, P. V., Hogden, J., Theiler, J., Xue, D., and Lookman, T. (2016). Accelerated search for materials with targeted properties by adaptive design. *Nature communications*, 7(1):1–9.

A Additional Background Details

A.1 Regret

Typically, in a multi-armed bandit problem (Lattimore and Szepesvári, 2020), an algorithm encounters K unknown probability distributions of rewards. The algorithm can only known more about it by sampling the distribution (or often referred as *arm*). Now, the goal of a bandit algorithm is to maximise total sum of accumulated rewards, i.e. $\sum_{t=1}^{T} R_t$, given a time horizon T. Typically, what we aim to maximize is the expectation of accumulated rewards, i.e. $\mathbb{E}[\sum_{t=1}^{T} R_t]$.

If we want to maximize the value of f for a point x in the subdomain covered by a node (h, i), it becomes equivalent to maximising the total obtained reward through its children. Thus, multi-armed bandit algorithms are deployed in HOO in order to maximize the value of an objective function, while the tree-based partition is given.

There is an alternative way of formulating the goal of a bandit, i.e. minimizing deviation of the expected accumulated reward $\mathbb{E}[\sum_{t=1}^{T} R_t]$ from the maximal achievable reward Tf^*. This is called *expected cumulative regret* or simply *regret*.

$$\begin{aligned}\mathbb{E}[\mathrm{Reg}_T] &= T\mu^* - \sum_{a=1}^{K} \mathbb{E}_\pi\left[N_T^a\right]\mu_a \\ &= \sum_{a=1}^{K} \mathbb{E}_\pi\left[N_T^a\right](\mu^* - \mu_a), \qquad \text{since, } T = \sum_{a=1}^{K} \mathbb{E}_\pi\left[N_T^a\right].\end{aligned}$$

Here, K is the number of arms, $\mathbb{E}_\pi\left[N_T^a\right]$ is the expected number of times the arm a is drawn, and $\mu^* - \mu_a$ is the expected suboptimality of arm a. Following the traditional analysis of hierarchical tree search algorithms (Munos, 2014), we are going to use regret as the measure of performance. Less is the regret better is the performance of the algorithm. If the upper bound on regret grows sublinearly with horizon T, it means that the error incurred by corresponding algorithm asymptotically vanishes.

The other performance metric relevant to a black-box optimization algorithm is the *error* incurred at any time t:

$$r_t = \max_x f(x) - f(x_t) = f^* - f(x_t). \tag{10}$$

r_t is also termed as *simple regret* in bandit literature. This metric is different than regret but in case of HOO, their expected values are closely related. If we choose the state x_t uniformly randomly from all the states observed till time T, the expected value of error (or simple regret) becomes

$$\mathbb{E}[r_t] = \mathbb{E}[f^* - f(x_t)] = \frac{1}{T}\sum_{i=1}^{t}[f^* - f(x_i)] = \frac{1}{T}\mathrm{Reg}_T \tag{11}$$

Later, we will use this relation to convert the traditional regret bound originating from the bandit algorithms used in PCTS to the expected error or the expected simple regret.

A.2 Optimistic Algorithms for Multi-armed Bandits

In both finite and continuous armed bandits, one of the successful paradigm is to design algorithms with Optimism in Front of Uncertainty (OFU) principle. OFU-type of algorithms, such as Upper Confidence Bound (UCB), UCB-σ, UCB-V etc., at each step t computes an optimistic value $B_{i,t}$ for each arm $i \in \mathcal{A}$ and chooses the arm with the maximum optimistic value. This optimistic value depends on the mean reward obtained from the arm, the number of pulls on the arm, and sometimes other carefully chosen statistics (e.g. variance and range) of rewards generated by the arm.

For example, for UCB Auer et al. (2002), $B_{i,t}^{\mathrm{UCB}} \triangleq \hat{\mu}_{i,t} + \sqrt{\frac{2\log t}{T_i(t-1)}}$. Here, $T_i(t-1)$ is the number of times arm i is played till time step t and $\hat{\mu}_{i,t}$ is the empirical mean of rewards obtained from arm i till time t. UCB does not consider any noise added to the rewards.

If the reward arrives with a noise of known variance σ^2, the UCB bound can be adapted to create UCB-σ Auer et al. (2007). For UCB-σ, $B_{i,t}^{\mathrm{UCB}-\sigma} \triangleq \hat{\mu}_{i,t} + \sqrt{\frac{2\sigma^2\log t}{T_i(t-1)}}$. Since in most of the cases

the noise variance σ^2 is not known, UCB-V is designed to adapt for noise with unknown variance and bounded range $[0, b]$. In this case, the noise variance for arm i is estimated at each step t as $\hat{\sigma}^2_{i,t} \triangleq \sum_{t=1}^{T} \mathbb{1}[a_t = i](f(x_t) - \hat{\mu}^i_t)^2$. Following that, the effective noise variance is used to define the optimistic value of UCB-V Audibert et al. (2007) as $B^{\mathrm{UCB-V}}_{i,t} \triangleq \hat{\mu}_{i,t} + \sqrt{\frac{2\hat{\sigma}^2_{i,t} \log t}{T_i(t-1)}} + c\frac{3b \log t}{T_i(t-1)}$. Here, $c > 0$ is some tunable parameter, which we consider 1 throughout our analysis. Interested practitioner may like to experiment with it.

The triumph of the OFU-type of algorithms is due to the fact that they achieve $O(\log T)$ regret bounds after T time steps, which is the optimal achievable regret according to the problem-dependent lower bound on regret of stochastic bandits (Lattimore and Szepesvári, 2020, Chapter 16). For further details on OFU algorithms, we refer to (Lattimore and Szepesvári, 2020, Chapters 7-10).

In last decade, the bandit community has been interested to adapt these OFU-type algorithms to delayed setup. Joulani et al. (2013, 2016); Vernade et al. (2017); Pike-Burke et al. (2018); Li et al. (2019) have extended some OFU-type algorithms, such as UCB and KL-UCB, to different delayed feedback settings, such as observable delay, arm-dependent delay etc.

B Proof Details

B.1 Assumptions: Structural Requirements of Optimistic Tree Search

In order to proof convergence of UDO, we oblige by the assumptions regarding theoretical analysis of HOO (Bubeck et al., 2011). In this section, we elaborate them.

Contracting Hierarchical Partition $\mathcal{T}$. The hierarchical optimistic tree search (HOO) or $\mathcal{X}$-armed bandits rely on existence of a hierarchical partitioning $\mathcal{T}$ of the domain $\mathcal{X}$. Let us represent the interval covered by the l-th node at depth h as $X_{h,i}$, where $l \in \{1, \dots, 2^h\}$ and $h \in \{0, \dots, H\}$. Then, we can define the corresponding hierarchical tree inducing the partition as $\mathcal{T} \triangleq \{X_{h,l}\}_{h,l=0,1}^{H,2^h}$. We observe that

$$X_{(0,1)} = \mathcal{X},$$
$$X_{(h,l)} = \cup_{j=0}^{K-1} X_{(h+1,Kl-j)} \cup X_{(h+1,Kl+j)},$$

where K is the maximum number of children of a node in this tree.

The specific value obtained at that node is denoted as $x_{h,i}$. Let us also assume that the domain of f, say $\mathcal{X} \subset \mathbb{R}^D$, has a dissimilarity measure or semi-metric ℓ that can quantify difference in output due to two inputs.

Assumption 2 (Hierarchical Partition with Decreasing Diameter and Shape)**.**

1.1 Decreasing diameters. *There exists a decreasing sequence $\delta(h) > 0$ and constant $\nu_1 > 0$ such that*

$$\operatorname{diam}(X_{h,i}) \triangleq \max_{x \in X_{h,i}} \ell(x_{h,i}, x) \leq \nu_1 \delta(h), \tag{12}$$

for any depth $h \geq 0$, for any interval $X_{h,i}$, and for all $i = 1, \dots, 2^h$. For simplicity, we consider that $\delta(h) = \rho^h$ for some $\rho \in (0, 1)$.

1.2 Regularity of the intervals. *There exists a constant $\nu_2 > 0$ such that for any depth $h \geq 0$, every interval $X_{h,i}$ contains at least a ball $\mathcal{B}_{h,i}$ of radius $\nu_2 \rho^h$ and center $x_{h,i}$ in it. Since the tree creates a partition at any given depth h, $\mathcal{B}_{h,i} \cap \mathcal{B}_{h,i} = \emptyset$ for all $1 \leq i < j \leq 2^h$.*

Global Smoothness of f. The other condition that we need to prove convergence of HOO to a global optimum is smoothness of f around the optimum, say x^*. This is often referred as weak Lipschitz property.

Assumption 1 (Weak Lipschitzness of f)**.** *For all $x, y \in \mathcal{X}$, f satisfies*

$$f^* - f(y) \leq f^* - f(x) + \max\{f^* - f(x), \ell(x, y)\}, \tag{13}$$

where f^* is the optimal value of f achieved at a global optimum x^*.

This assumption holds true if

1. either $f(x) - f(y) \leq \ell(x, y)$ and $f^* - f(x) \leq \max_y \ell(x, y)$,
2. or $f(x) - f(y) \leq f^* - f(x)$ and $f^* - f(x) \geq \max_y \ell(x, y)$.

This property basically implies that there is no sudden drop or jump in performance f around the optimal point x^*. Weak Lipschitzness can hold even for discontinuous functions. Thus, it widens applicability of HOO's analysis to more general performance metrics and configuration spaces in comparison with algorithms that explicitly need gradients or smoothness in some form.

B.2 Regret of PCTS with Delayed Feedback

Let us define a few quantities before proceeding to the proofs.

1. $G_t = \sum_{s=1}^{t-1} \mathbb{1}\{s + \tau_s \geq t\}$ i.e. the number of missing feedbacks when the forecaster chooses the next action at time t.
2. $G_t^* = \max_{1 \leq s \leq t} G_t$ the maximum number of feedbacks not observed till time t. Note that it is τ_{const} for constant delay.
3. $G_{i,t}$ which is the number of missing feedbacks for action i at time t.
4. $T_i(t)$ which is the number of reward samples observed from arm i at time t in a non-delayed setting.
5. $S_i(t)$ which is the number of reward samples observed from arm i at time t in a delayed setting. Note that $T_i(t) = S_i(t) + G_{i,t}$.

On the other hand, we can describe the action selection process of any UCB-like optimistic bandit algorithm as:

$$a_t = \arg\max_{i \in \mathcal{A}} B_{i,s,t}.$$

Here, $B_{i,s,t}$ is the optimistic upper confidence bound for action i at time t, and s is the number of reward samples obtained from arm i till time t. For example, in case of UCB1 (Auer et al., 2002),

$$B_{i,s,t} = \hat{\mu}_{i,s} + \sqrt{\frac{2 \log t}{s}} = \frac{1}{s} \sum_{j=1}^{s} r_{i,j} + \sqrt{\frac{2 \log t}{s}}. \tag{14}$$

For non-delayed setting, $s = T_i(t-1)$ and for delayed setting $s = S_i(t-1)$.

Thus, given an aforementioned UCB-like optimistic bandit algorithm, the leaf node (h_t, j_t) selected by PCTS algorithm at time t is

$$(h_t, l_t) \triangleq \arg\max_{(h,l) \in \mathcal{T}_t} B^{\min}_{(h,l)}(t) \triangleq \arg\max_{(h,l) \in \mathcal{T}_t} \min\{B_{(h,l),s,t} + \nu_1 \rho^h, \max_{(h',l') \in C(h,l)} B^{\min}_{(h,l)}(t)\}. \tag{15}$$

Given these notations and definitions, now we elaborate the proof sketch of PCTS following that of HOO and show the generic technique to incorporate regret in it. The analysis described in this section, we assume that the smoothness parameters ν_1 and ρ are known. We loosen this assumption in the following section.

B.2.1 Generic Proof Sketch: From Regret to the Number of Visits to a Suboptimal Node

Theorem 5. *Let us consider that the expected objective function f satisfies Assumption 1, and its $4\nu_1/\nu_2$-near-optimality dimension is $d > 0$. Then, under Assumption 2 and for any $d' > d$, PCTS algorithm uses a bandit algorithm MAB for node selection will achieve expected regret*

$$\begin{aligned} \mathbb{E}[\mathrm{Reg}_T] \leq\ & 4C\nu_1\nu_2^{-d'} \sum_{h=0}^{H-1} (\delta(h))^{(1-d')} + 4\nu_1 \delta(H) T \\ & + 8C\nu_1\nu_2^{-d'} \sum_{h=1}^{H} (\delta(h-1))^{(1-d')} \times (U(T, \tau, \delta(h))). \end{aligned} \tag{16}$$

Here, $U(T, \tau, \delta(h))$ is the upper bound on number of visits to the $2\nu_1\delta(h)$-suboptimal nodes at depth $h > 0$ by BANDIT.

Proof. Now, we proceed with the regret analysis of PCTS which is essentially similar with the analysis of the HOO while we try to accommodate the delayed feedbacks in it. The proof can be divided into three steps: a) regret decomposition to suboptimal and optimal node exploration, b) bounding the number of times the suboptimal nodes and the optimal nodes reached through the suboptimal nodes, c) bringing in the effect of delay and UCB-like algorithms in above too and merging them.

Step 1: Regret Decomposition. Let us divide the nodes of the Monte Carlo tree $\mathcal{T}$, which can possibly grow infinitely, in three categories such that $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \mathcal{T}_3$.

In order to define, these subcategories first let us define the 'optimal' nodes I and 'suboptimal' nodes reached by playing optimal nodes J.

Let us denote the set of $2\nu_1\delta(h)$-optimal nodes at depth h ($h \in \mathbb{Z}_{\geq 0}$) as I_h, i.e.

$$I_h \triangleq \{(h,i) | f^*_{h,i} \geq f^* - 2\nu_1\delta(h)\}.$$

Thus, the set of all optimal nodes in the MC-tree is $I = \bigcup_h I_h$. Note that, the root node has $(h,i) = (0,1)$ and $I_0 = \{(0,1)\}$.

Let us denote the nodes $J = \bigcup_h J_h$ who are not essentially in I but whose parents are in I. These are the suboptimal nodes to be reached through playing the optimal nodes. As the root node is in I_0, we can observe that all the $2\nu_1\delta(h)$-suboptimal nodes (for all $h > 0$) in the tree are children of either nodes in I or those in J.

Given these definitions, we can define now the three subcategories of nodes that we mentioned earlier.

$\mathcal{T}_1$: Set of all optimal nodes in the tree $I_H = \bigcup_{h=0}^{H} I_h$.

$\mathcal{T}_2$: All the nodes which are descendants[4] of nodes in I_H.

$\mathcal{T}_3$: All the nodes which are descendants of nodes in $J_H = \bigcup_{h=1}^{H} J_h$.

Now, we decompose the expected regret in three components corresponding to each of these three categories:

$$\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T] &= \mathbb{E}[\sum_{t=1}^{T}(f^* - f(X_t))] \\
&= \mathbb{E}[\sum_{t=1}^{T}(f^* - f(X_t))]\mathbb{1}[(H_t, I_t) \in \mathcal{T}_1] + \mathbb{E}[\sum_{t=1}^{T}(f^* - f(X_t))]\mathbb{1}[(H_t, I_t) \in \mathcal{T}_2] \\
&\qquad + \mathbb{E}[\sum_{t=1}^{T}(f^* - f(X_t))]\mathbb{1}[(H_t, I_t) \in \mathcal{T}_3] \\
&\triangleq \mathbb{E}[\mathrm{Reg}_{T,1}] + \mathbb{E}[\mathrm{Reg}_{T,2}] + \mathbb{E}[\mathrm{Reg}_{T,3}] \qquad (17)
\end{aligned}$$

By Lemma 3 in (Bubeck et al., 2011), we obtain that

$$\mathbb{E}[\mathrm{Reg}_{T,1}] \leq \sum_{h=0}^{H-1} 4\nu_1\delta(h)|I_h|. \qquad (18)$$

and,

$$\mathbb{E}[\mathrm{Reg}_{T,2}] \leq 4\nu_1\delta(H)T. \qquad (19)$$

Now, for $\mathcal{T}_3$, we observe that parent of any element in J_h is in I_{h-1}. Thus, the region covered by these nodes is a subset of the region covered by I_{h-1} (Assumption 2 and 1). Thus, we obtain

$$\begin{aligned}
\mathbb{E}[\mathrm{Reg}_{T,3}] &\leq \sum_{h=1}^{H} 4\nu_1\delta(h-1) \sum_{i:(h,i)\in J_h} \mathbb{E}[S_{h,i}(T)] \qquad (20) \\
&\leq \sum_{h=1}^{H} 4\nu_1\delta(h-1)|J_h| \max_{(h,i)\in J_h} \mathbb{E}[S_{h,i}(T)] \qquad (21)
\end{aligned}$$

[4]Descendants of a node include the node itself.

$$\leq \sum_{h=1}^{H} 8\nu_1\delta(h-1)|I_{h-1}| \max_{(h,i)\in J_h} \mathbb{E}[S_{h,i}(T)]. \tag{22}$$

The last inequality holds as the parents of nodes of J_h are in I_{h-1}, and by the way the tree grows $|J_h| \leq 2|I_{h-1}|$ for any $h \geq 1$.

Step 2: Bounding the Number of Optimal Nodes and Direct Descendants of Optimal Nodes. Now, we want to bound two things. Firstly, the size of the optimal nodes at level h of the MC-tree, i.e. $|I_h|$. Secondly, the expected number of times the nodes in J_h are visited, i.e. $\mathbb{E}[S_{h,i}(t)|(h,i) \in J_h]$.

From (Bubeck et al., 2011), we observe that bounding the first quantity, i.e. $|I_h|$, is independent of the MC-tree dynamics under Assumption 2. Thus, we get

$$|I_h| \leq C(\nu_2\delta(h))^{-d'} \text{ for some } d' \geq d. \tag{23}$$

The second quantity depends on the UCB-like algorithm used for next action/node selection and also the delay model. In all these cases, we can show that

$$\mathbb{E}[S_{h,i}(t)|(h,i) \in J_h] \leq U(t,\tau,\delta(h)). \tag{24}$$

In corresponding sections, we prove the specific forms of $U(t,\tau,\delta(h))$ for UCB-like algorithms and delay models.

Step 3: Merging the Effects of Delay and Suboptimal Node Plays. If we merge all the aforementioned results, we obtain an upper bound on the total regret of the PCTS framework.

$$\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T] &\leq \sum_{h=0}^{H-1} 4\nu_1\delta(h)|I_h| + 4\nu_1\delta(H)T + \sum_{h=1}^{H} 8\nu_1\delta(h-1)|I_{h-1}| \max_{(h,i)\in J_h} \mathbb{E}[S_{h,i}(t)] \\
&\leq \sum_{h=0}^{H-1} 4\nu_1\delta(h) \times C(\nu_2\delta(h))^{-d'} \\
&\quad + 4\nu_1\delta(H)T \\
&\quad + \sum_{h=1}^{H} 8\nu_1\delta(h-1) \times C(\nu_2\delta(h-1))^{-d'} \times (U(T,\tau,\delta(h))) \\
&\leq 4C\nu_1\nu_2^{-d'} \sum_{h=0}^{H-1} (\delta(h))^{(1-d')} + 4\nu_1\delta(H)T \\
&\quad + 8C\nu_1\nu_2^{-d'} \sum_{h=1}^{H} (\delta(h-1))^{(1-d')} \times (U(T,\tau,\delta(h))).
\end{aligned} \tag{25}$$

Thus, we get the desired regret bound for PCTS with observable stochastic delays in feedbacks and PCTS algorithm with any efficient bandit algorithm MAB for node selection. □

B.2.2 Expected Visits of a Node (h,i) under Delay

Lemma 1. *Let us consider (h,i) is a $2\nu_1\delta(h)$-suboptimal node in J_h and there's an observable stochastic delay $G_{(h,i),t}$ while receiving the feedback. Then for any integer $u \geq 0$,*

$$\begin{aligned}
\mathbb{E}[S_{h,i}(T)] \leq u + \mathbb{E}[G^*_{(h,i),T}] + \sum_{t=u+1}^{T} (&\mathbb{P}[S_{h,i}(t) > u \wedge B_{(h,i),S_{h,i},t} + \nu_1\delta(h) > f^*] \\
&+ \sum_{s=1}^{t} \mathbb{P}[B_{(h,i),S_{h,i},t} + \nu_1\delta(h) \leq f^*]).
\end{aligned} \tag{26}$$

Proof. For the undelayed feedback, the number of times a $2\nu_1\delta(h)$-suboptimal node (h,i) is visited can be written as

$$T_{h,i}(T) = \sum_{t=1}^{T} \mathbb{1}[(H_t, I_t) \in \text{Descendant}(h,i) \wedge T_{h,i}(t) \leq u]$$

$$+\sum_{t=1}^{T}\mathbb{1}[(H_t,I_t)\in\text{Descendant}(h,i)\wedge T_{h,i}(t)>u]$$
$$\leq u+\sum_{t=u+1}^{T}\mathbb{1}[(H_t,I_t)\in\text{Descendant}(h,i)\wedge T_{h,i}(t)>u].$$

Now, to shift this analysis to the observable stochastic delayed feedback model, we replace $T_{h,i}(t)$ with $S_{h,i}(t)+G_{h,i}(t)$ and u with $u'+G^*_{(h,i),T}$. Thus, we get

$$S_{h,i}(T)\leq u'+G^*_{(h,i),T}+\sum_{t=u'+1}^{T}\mathbb{1}[\underbrace{(H_t,I_t)\in\text{Descendant}(h,i)}_{\text{Event }E_1}\wedge S_{h,i}(t)>u'].$$

Now, in order to understand the event E_1, let us consider that the path from the root node $(0,1)$ to node (h,i) passes through an optimal node (k,i^*_k) last at depth $k\in[0,h-1]$ and does not visit an optimal node afterwards. Under this specification, the path toward the suboptimal node (h,i) is chosen than the next optimal node $(k+1,i^*_{k+1})$ if

$$B_{(h,i),S_{h,i},t}+\nu_1\delta(h)\geq B_{(k+1,i^*_{k+1}),S_{k+1,i^*_{k+1}},t}.$$

This event is satisfied if $E_2: B_{(h,i),S_{h,i},t}+\nu_1\delta(h)\geq f^*$ and $E'_{k+1}: f^*\geq B_{(k+1,i^*_{k+1}),S_{k+1,i^*_{k+1}},t}$ hold true. Thus, $E_1\subset E_2\cup E'_{k+1}$.

We also observe that $E'_{k+1}\subset E_{k+1}\cup E'_{k+2}$, where $E_{k+1}: B_{(k+1,i^*_{k+1}),S_{(k+1,i^*_{k+1})},t}+\nu_1\delta(k+1)\leq f^*$ and $E'_{k+2}: f^*\geq B_{(k+2,i^*_{k+1}),S_{k+1,i^*_{k+1}},t}$. Iterating similar argument from $k+1$ to t, we get $E'_{k+1}\subset\cup_{s=k+1}^{t-1}E_s$. The induction stops at $t-1$ because the node (t,i^*_t) is yet not visited and thus, the upper confidence value assigned to it is $+\infty$, which is not bounded by f^* for sure.

Hence, $E_1\subset E_2\cup(\cup_{s=k+1}^{t-1}E_s)$, and number of visits to node (h,i) under delayed feedback

$$S_{h,i}(T)\leq u'+G^*_{(h,i),T}+\sum_{t=u'+1}^{T}\mathbb{1}[(E_2\cup(\cup_{s=k+1}^{t-1}E_s))\wedge S_{h,i}(t)>u']$$
$$\leq u'+G^*_{(h,i),T}+\sum_{t=u'+1}^{T}\mathbb{1}[(E_2\wedge S_{h,i}(t)>u')\cup(\cup_{s=k+1}^{t-1}E_s)].$$

Thus,

$$\begin{aligned}\mathbb{E}[S_{h,i}(T)] &\leq u'+\mathbb{E}[G^*_{(h,i),T}]+\sum_{t=u'+1}^{T}\mathbb{P}[(E_2\wedge S_{h,i}(t)>u')\cup(\cup_{s=k+1}^{t-1}E_s)]\\ &\underset{Union\ bound}{\leq} u'+\mathbb{E}[G^*_{(h,i),T}]+\sum_{t=u'+1}^{T}\left(\mathbb{P}[E_2\wedge S_{h,i}(t)>u']+\sum_{s=k+1}^{t-1}\mathbb{P}[E_s]\right)\\ &\leq u'+\mathbb{E}[G^*_{(h,i),T}]+\sum_{t=u'+1}^{T}\left(\mathbb{P}[E_2\wedge S_{h,i}(t)>u']+\sum_{s=1}^{t-1}\mathbb{P}[E_s]\right).\end{aligned}$$

□

B.2.3 Specification for Delayed-UCB1

Up to this point the analysis is independent of the choice of the bandit algorithm for node selection. For an example, let us choose the bandit algorithm to be Delayed-UCB1 (Joulani et al., 2013). From Delayed-UCB1, we obtain that

$$B_{(h,i),S_{(h,i)}(t-1),t}\triangleq\hat{\mu}_{(h,i),S_{(h,i)}(t-1)}+\sqrt{\frac{2\log t}{S_{(h,i)}(t-1)}}\tag{27}$$

Under this decision rule, we prove the upper bound on the expected number of visits to any $2\nu_1\delta(h)$-suboptimal node (h,i).

Corollary 3. *Let us consider (h,i) is a $2\nu_1\delta(h)$-suboptimal node in J_h and there's an observable stochastic delay $G_{(h,i),t}$ in feedback with expectation bound τ. If we use Delayed-UCB1 for node selection at any time t, we obtain that*

$$U(T,\tau,\delta(h)) = \frac{8\ln T}{(\nu_1\delta(h))^2} + \tau + 4. \tag{28}$$

Proof. Now, if we choose $u' = \frac{8\ln T}{(\nu_1\delta(h))^2}$, using Hoeffding's concentration inequalities, and Assumptions 2 and 1, we get from Lemma 1

$$\begin{aligned}
\mathbb{E}[S_{h,i}(T)] &\leq \frac{8\ln T}{(\nu_1\delta(h))^2} + \mathbb{E}[G^*_{(h,i),T}] + \sum_{t=u'+1}^{T}\left(\mathbb{P}[E_2 \wedge S_{h,i}(t) > u'] + \sum_{s=1}^{t-1}\mathbb{P}[E_s]\right) \\
&\leq \frac{8\ln T}{(\nu_1\delta(h))^2} + \mathbb{E}[G^*_{(h,i),T}] + \sum_{t=u'+1}^{T}\left(tT^{-4} + \sum_{s=1}^{t-1}t^{-3}\right) \\
&\leq \frac{8\ln T}{(\nu_1\delta(h))^2} + \mathbb{E}[G^*_{(h,i),T}] + 4 \\
&\leq \frac{8\ln T}{(\nu_1\delta(h))^2} + \tau + 4.
\end{aligned}$$

□

Theorem 2 (Regret of PCTS + DUCB1)**.** *Let us consider that the expected objective function f satisfies Assumption 1, and its $4\nu_1/\nu_2$-near-optimality dimension is $d > 0$. Then, under Assumption 2 and for any $d' > d$, PCTS algorithm using Delayed-UCB1 (DUCB1) achieves expected regret*

$$\mathbb{E}[\mathrm{Reg}_T] = O\left(T^{1-\frac{1}{d'+2}}(\ln T)^{\frac{1}{d'+2}}\left(1+\frac{\tau}{\ln T}\right)^{\frac{1}{d'+2}}\right), \tag{29}$$

and expected simple regret

$$\epsilon_T = O\left(\left(\frac{\ln T}{T}\right)^{\frac{1}{d'+2}}\left(1+\frac{\tau}{\ln T}\right)^{\frac{1}{d'+2}}\right), \tag{30}$$

where the expected delay is upper bounded by τ.

Proof. From Theorem 5 and Corollary 3, we get

$$\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T] &\leq 4C\nu_1\nu_2^{-d'}\sum_{h=0}^{H-1}(\delta(h))^{(1-d')} + 4\nu_1\delta(H)T \\
&\quad + 8C\nu_1\nu_2^{-d'}\sum_{h=1}^{H}(\delta(h-1))^{(1-d')} \times \left(\frac{8\ln T}{(\nu_1\delta(h))^2} + \tau + 4\right) \\
&\leq 4C\nu_1\nu_2^{-d'}\sum_{h=0}^{H-1}(\delta(h))^{(1-d')}(2\tau+9) + 4\nu_1\delta(H)T \\
&\quad + 64C\nu_1^{-1}\nu_2^{-d'}\sum_{h=1}^{H}(\delta(h-1))^{(1-d')}\delta(h)^{-2} \times \ln T.
\end{aligned}$$

For simplicity, we represent the decreasing diameter as $\delta(h) = \rho^h$ for some $\rho \in (0,1)$. Thus,

$$\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T] &\leq 4C\nu_1\nu_2^{-d'}\sum_{h=0}^{H-1}\rho^{h(1-d')}(2\tau+9) + 4\nu_1\rho^H T \\
&\quad + 64C\nu_1^{-1}\nu_2^{-d'}\sum_{h=1}^{H}\rho^{h(1-d')-(1-d')}\rho^{-2h} \times \ln T
\end{aligned}$$

$$
\begin{aligned}
&\leq 4C\nu_1\nu_2^{-d'}\sum_{h=0}^{H-1}\rho^{h(1-d')}(2\tau+9)+4\nu_1\rho^H T\\
&+64C\nu_1^{-1}\nu_2^{-d'}\sum_{h=1}^{H}\rho^{-h(1+d')-(1-d')}\times\ln T\\
&=O(\sum_{h=0}^{H-1}\rho^{h(1-d')}\tau+\sum_{h=0}^{H-1}\rho^{-h(1+d')}\ln T+\rho^H T) \qquad (31)
\end{aligned}
$$

Since $\rho\in(0,1)$, we get

$$
\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T]&=O(\sum_{h=0}^{H-1}\rho^{-h(1+d')}(\tau+\ln T)+\rho^H T)\\
&=O(\rho^{-H(1+d')}(\tau+\ln T)+\rho^H T)
\end{aligned}
$$

By choosing a ρ such that $\rho^{-H(d'+2)}=\frac{T}{\tau+\ln T}$, we obtain

$$
\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T]&=O\left(\left(\frac{T}{\tau+\ln T}\right)^{\frac{d'+1}{d'+2}}(\tau+\ln T)+\left(\frac{\tau+\ln T}{T}\right)^{\frac{1}{d'+2}}T\right)\\
&=O\left(T^{1-\frac{1}{d'+2}}(\tau+\ln T)^{\frac{1}{d'+2}}\right)\\
&=O\left(T^{1-\frac{1}{d'+2}}(\ln T)^{\frac{1}{d'+2}}\left(1+\frac{\tau}{\ln T}\right)^{\frac{1}{d'+2}}\right).
\end{aligned}
$$

This trivially leads to the bound on simple regret, i.e. the expected error at each step, as $\epsilon_T=\frac{1}{T}\mathbb{E}[\mathrm{Reg}_T]$. □

Corollary 1 (Regret of PCTS + DUCB1 under Constant Delay)**.** *If the delay in feedback is constant, i.e. $\tau_{const}>0$, the expected regret of the PCTS algorithm using Delayed-UCB1 (DUCB1)*

$$
\mathbb{E}[\mathrm{Reg}_T]=O\left(T^{1-\frac{1}{d'+2}}(\ln T)^{\frac{1}{d'+2}}\left(1+\frac{\tau_{const}}{\ln T}\right)^{\frac{1}{d'+2}}\right), \qquad (32)
$$

and expected simple regret

$$
\epsilon_T=O\left(\left(\frac{\ln T}{T}\right)^{\frac{1}{d'+2}}\left(1+\frac{\tau_{const}}{\ln T}\right)^{\frac{1}{d'+2}}\right). \qquad (33)
$$

Proof. We don't need an upper bound on expected maximum delay in this setting. Since all the delays are τ_{const}, by default the expectation is exactly τ_{const}. Putting that in the derivation trivially provides us this results. □

B.2.4 Lower Bounds for Constant Delay Setting

Theorem 6.

B.3 Regret of PCTS with Delayed and Noisy Feedback

Typically, when we evaluate the objective function at any time step t, we obtain a noisy version of the function as feedback such that $\tilde{f}(X_t) = f(X_t) + \epsilon_t$. Here, ϵ_t is a noise sampled independently generated from a noise distribution $\mathcal{N}$. Till now, we did not explicitly consider the noise for the action selection step. In this section, we provide analysis for that for both known and unknown variance cases. In both of the cases, we assume that the noise has bounded mean and variance. This holds for the present setup as any noisy evaluation can be clipped in the range of the evaluations where we optimise the objective functions. And, we know that a bounded random variable is sub-Gaussian with bounded mean and variance.

B.3.1 Noise with Known Variance

Let us assume that the variance of the noise is known, say $\sigma^2 > 0$. In this case, the optimistic bounds are computed using a simple variant of delayed-UCB1, say delayed-UCB-σ^2 (Equation (27)), where

$$B_{(h,i),S_{(h,i)}(t-1),t} \triangleq \hat{\mu}_{(h,i),S_{(h,i)}(t-1)} + \sqrt{\frac{2\sigma^2 \log t}{S_{(h,i)}(t-1)}}. \tag{34}$$

Here, $\hat{\mu}_{(h,i),S_{(h,i)}(t-1)}$ is the empirical mean computed using noisy evaluations.

Using confidence bound of Equation (6), we obtain a modified version of the Corollary 3.

Corollary 4. *Let us consider (h,i) is a $2\nu_1\delta(h)$-suboptimal node in J_h, there's an observable stochastic delay $G_{(h,i),t}$ in feedback with expectation bound τ, and the variance of the added noise in evaluations is σ^2. If we use delayed-UCB-σ^2 (Equation (6)) for node selection at any time t, we obtain that*

$$U(T, \tau, \delta(h)) = \frac{8\sigma^2 \ln T}{(\nu_1\delta(h))^2} + \tau + 4. \tag{35}$$

Proof. Now, if we choose $u' = \frac{8\sigma^2 \ln T}{(\nu_1\delta(h))^2}$, similarly using Hoeffding's concentration inequalities, and Assumptions 2 and 1, we get from Lemma 1

$$\begin{aligned}
\mathbb{E}[S_{h,i}(T)] &\leq \frac{8\sigma^2 \ln T}{(\nu_1\delta(h))^2} + \mathbb{E}[G^*_{(h,i),T}] + \sum_{t=u'+1}^{T} \left(\mathbb{P}[E_2 \wedge S_{h,i}(t) > u'] + \sum_{s=1}^{t-1} \mathbb{P}[E_s] \right) \\
&\leq \frac{8\sigma^2 \ln T}{(\nu_1\delta(h))^2} + \mathbb{E}[G^*_{(h,i),T}] + \sum_{t=u'+1}^{T} \left(tT^{-4} + \sum_{s=1}^{t-1} t^{-3} \right) \\
&\leq \frac{8\sigma^2 \ln T}{(\nu_1\delta(h))^2} + \tau + 4.
\end{aligned}$$

□

Theorem 3 (Regret of PCTS + DUCB1σ)**.** *Let us assume that the variance of the noise in evaluations is σ^2 and is available to the algorithm. Then, under the same assumptions as Theorem 1 and upper bound on expected delay τ, PCTS using DUCB1σ achieves expected simple regret*

$$\epsilon_T = O\left(T^{-\frac{1}{d'+2}} \left((\sigma/\nu_1)^2 \ln T + \tau \right)^{\frac{1}{d+2}} \right). \tag{36}$$

Proof. Using the result of Corollary 4 in the proof schematic of Theorem 2, we obtain that PCTS algorithm using Equation (6) for node selection will achieve

$$\mathbb{E}[\mathrm{Reg}_T] = O(\rho^{-H(1+d')}(\tau + (\sigma/\nu_1)^2 \ln T) + \rho^H T)$$

By choosing a ρ such that $\rho^{-H(d'+2)} = \frac{T}{\tau + (\sigma/\nu_1)^2 \ln T}$, we obtain

$$\mathbb{E}[\mathrm{Reg}_T] = O\left(\left(\frac{T}{\tau + (\sigma/\nu_1)^2 \ln T} \right)^{\frac{d'+1}{d'+2}} (\tau + (\sigma/\nu_1)^2 \ln T) + \left(\frac{\tau + (\sigma/\nu_1)^2 \ln T}{T} \right)^{\frac{1}{d'+2}} T \right)$$

$$= O\left(T^{1-\frac{1}{d'+2}}(\tau + (\sigma/\nu_1)^2 \ln T)^{\frac{1}{d'+2}}\right)$$

Thus, for PCTS with noisy evaluations and confidence bounds of Equation (6), i.e. DUCB1σ, achieves a simple regret

$$\epsilon_T = O\left(T^{1-\frac{1}{d'+2}}(\tau + (\sigma/\nu_1)^2 \ln T)^{\frac{1}{d'+2}}\right)$$

where the expected delay is upper bounded by τ and σ^2 is the noise variance. □

B.3.2 Noise with Unknown Variance

If the variance of the noise is unknown, we have to estimate the noise variance empirically from evaluations. Given the evaluations $\tilde{f}(X_1), \ldots, \tilde{f}(X_T)$ and the delayed statistics $S_{(h,i)}(t-1)$ of node (h,i), the empirical noise variance at time t is

$$\hat{\sigma}^2_{(h,i),S_{(h,i)}(t-1)} \triangleq \frac{1}{S_{(h,i)}(t-1)} \sum_{j=1}^{S_{(h,i)}(t-1)} \left(\tilde{f}(X_j)\mathbb{1}[(H_j, I_j) = (h,i)] - \hat{\mu}_{(h,i),S_{(h,i)}(t-1)}\right)^2,$$

where empirical mean $\hat{\mu}_{(h,i),S_{(h,i)}(t-1)} \triangleq \frac{1}{S_{(h,i)}(t-1)} \sum_{j=1}^{S_{(h,i)}(t-1)} \tilde{f}(X_j)\mathbb{1}[(H_j, I_j) = (h,i)]$.

Using the empirical mean and variance of functional evaluations for each node (h,i), we now define a delayed-UCBV[5] confidence bound:

$$B_{(h,i),S_{(h,i)}(t-1),t} \triangleq \hat{\mu}_{(h,i),S_{(h,i)}(t-1)} + \sqrt{\frac{2\hat{\sigma}^2_{(h,i),S_{(h,i)}(t-1)} \log t}{S_{(h,i)}(t-1)}} + \frac{3b \log t}{S_{(h,i)}(t-1)}. \tag{37}$$

Corollary 5. *Let us consider (h,i) is a $2\nu_1\delta(h)$-suboptimal node in J_h, there's an observable stochastic delay $G_{(h,i),t}$ in feedback with expectation bound τ, and the variance of the added noise in evaluations is σ^2. If we use Delayed-UCBV (Equation (8)) for node selection at any time t, we obtain that*

$$U(T, \tau, \delta(h)) = c\left(\frac{\sigma^2}{(\nu_1\delta(h))^2} + \frac{2b}{(\nu_1\delta(h))}\right) \ln T + \tau. \tag{38}$$

Proof. Now, if we choose $u' = 1 + 8c_1\left(\frac{\sigma^2}{(\nu_1\delta(h))^2} + \frac{2b}{(\nu_1\delta(h))}\right) \ln T$,[6] we get from Lemma 1

$$\mathbb{E}[S_{h,i}(T)] \leq 1 + 8c_1\left(\frac{\sigma^2}{(\nu_1\delta(h))^2} + \frac{2b}{(\nu_1\delta(h))}\right) \ln T + \mathbb{E}[G^*_{(h,i),T}] + \sum_{t=u'+1}^{T}\left(\mathbb{P}[E_2 \wedge S_{h,i}(t) > u'] + \sum_{s=1}^{t-1}\mathbb{P}[E_s]\right)$$

By Theorem 3 in (Audibert et al., 2007), we get $\sum_{s=1}^{t-1}\mathbb{P}[E_s] \leq \exp[-c_1 \ln T]\left(\frac{24\sigma^2}{(\nu_1\delta(h))^2} + \frac{4b}{(\nu_1\delta(h))}\right)$. This is a consequence of the empirical Bernstein's inequality. Thus,

$$\sum_{t=u'+1}^{T}\sum_{s=1}^{t-1}\mathbb{P}[E_s] \leq Te^{-c_1 \ln T}\left(\frac{24\sigma^2}{(\nu_1\delta(h))^2} + \frac{4b}{(\nu_1\delta(h))}\right) \leq 0.21\left(\frac{24\sigma^2}{(\nu_1\delta(h))^2} + \frac{4b}{(\nu_1\delta(h))}\right) \ln T,$$

for $c_1 = 1.2$. By Theorem 1 in (Audibert et al., 2007), we get

$$\sum_{t=u'+1}^{T}\mathbb{P}[E_2 \wedge S_{h,i}(t) > u'] \leq \sum_{t=u'+1}^{T}\beta(c_1 \ln s, s) \leq 0.07\frac{2b}{\nu_1\delta(h)} \ln T,$$

for $c_1 = 1.2$. Combining these three equations together yield the desired result with $c = 10$. □

[5] UCB-V without delay is proposed in (Audibert et al., 2007).

[6] We fix $c_1 = 1.2$.

Theorem 4 (Regret of PCTS + DUCBV with Unknown Noise)**.** *Let us consider that the expected objective function f satisfies Assumption 1, and its $4\nu_1/\nu_2$-near-optimality dimension is $d > 0$. Let us also assume that the variance of the noise in evaluations is σ^2 which is unknown to the algorithm. Then, under Assumption 2 and for any $d' > d$, PCTS algorithm using DUCBV achieves expected regret*

$$\mathbb{E}[\mathrm{Reg}_T] = O\left(T^{1-\frac{1}{d'+2}}(\tau + ((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T)^{\frac{1}{d'+2}}\right) \tag{39}$$

and expected simple regret

$$\epsilon_T = O\left(T^{-\frac{1}{d'+2}}(\tau + ((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T)^{\frac{1}{d'+2}}\right) \tag{40}$$

where the expected delay is upper bounded by τ.

Proof. From Theorem 5 and Corollary 3, we get

$$\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T] &\leq 4C\nu_1\nu_2^{-d'}\sum_{h=0}^{H-1}(\delta(h))^{(1-d')} + 4\nu_1\delta(H)T \\
&+ 8C\nu_1\nu_2^{-d'}\sum_{h=1}^{H}(\delta(h-1))^{(1-d')} \times \left(\frac{10\sigma^2\ln T}{(\nu_1\delta(h))^2} + \frac{20b\ln T}{(\nu_1\delta(h))} + \tau\right) \\
&\leq 4C\nu_1\nu_2^{-d'}\sum_{h=0}^{H-1}(\delta(h))^{(1-d')}(2\tau) + 4\nu_1\delta(H)T \\
&+ (80\sigma^2)C\nu_1^{-1}\nu_2^{-d'}\sum_{h=1}^{H}(\delta(h-1))^{(1-d')}\delta(h)^{-2} \times \ln T \\
&+ (160b)C\nu_2^{-d'}\sum_{h=1}^{H}(\delta(h-1))^{(1-d')}\delta(h)^{-1} \times \ln T.
\end{aligned}$$

For simplicity, we represent the decreasing diameter as $\delta(h) = \rho^h$ for some $\rho \in (0,1)$. Thus,

$$\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T] &\leq 4C\nu_1\nu_2^{-d'}\sum_{h=0}^{H-1}\rho^{h(1-d')}(2\tau) + 4\nu_1\rho^H T \\
&+ (80\sigma^2)C\nu_1^{-1}\nu_2^{-d'}\sum_{h=1}^{H}\rho^{h(1-d')-(1-d')}\rho^{-2h} \times \ln T + (160b)C\nu_2^{-d'}\sum_{h=1}^{H}\rho^{h(1-d')-(1-d')}\rho^{-h} \times \ln T \\
&\leq 4C\nu_1\nu_2^{-d'}\sum_{h=0}^{H-1}\rho^{h(1-d')}(2\tau) + 4\nu_1\rho^H T \\
&+ (80\sigma^2)C\nu_1^{-1}\nu_2^{-d'}\sum_{h=1}^{H}\rho^{-h(1+d')-(1-d')} \times \ln T + (160b)C\nu_2^{-d'}\sum_{h=1}^{H}\rho^{-hd'-(1-d')} \times \ln T \\
&= O\left(\sum_{h=0}^{H-1}\rho^{h(1-d')}\tau + (\sigma/\nu_1)^2\sum_{h=0}^{H-1}\rho^{-h(1+d')}\ln T + 2b/\nu_1\sum_{h=0}^{H-1}\rho^{-hd'}\ln T + \rho^H T\right) \\
&= O\left(\sum_{h=0}^{H-1}\rho^{-h(1+d')}(\tau + (\sigma/\nu_1)^2\ln T + 2b/\nu_1\ln T) + \rho^H T\right), \text{since } \rho \in (0,1) \\
&= O\left(\rho^{-H(1+d')}(\tau + ((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T) + \rho^H T\right)
\end{aligned}$$

By choosing a ρ such that $\rho^{-H(d'+2)} = \frac{T}{\tau + ((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T}$, we obtain

$$\mathbb{E}[\mathrm{Reg}_T] = O\left(\left(\frac{T}{\tau + ((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T}\right)^{\frac{d'+1}{d'+2}}(\tau + (2b/\nu_1 + (\sigma/\nu_1)^2)\ln T) + \left(\frac{\tau + ((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T}{T}\right)^{\frac{1}{d'+2}} T\right.$$

$$= O\left(T^{1-\frac{1}{d'+2}}(\tau + ((\sigma/\nu_1)^2 + 2b/\nu_1)\ln T)^{\frac{1}{d'+2}}\right)$$

This trivially leads to the bound on simple regret, i.e. the expected error at each step, as $\epsilon_T = \frac{1}{T}\mathbb{E}[\text{Reg}_T]$

□

B.4 Regret of PCTS with Delayed, Noisy, and Multi-fidelity (DNF) Feedback

Now, let us consider that we don't only have a delayed and noisy functional evaluator at each step but also the evaluator has different fidelity at each level h of the MC tree. This setup of multi-fidelity MCTS without unknown noise and delay was first considered in (Sen et al., 2019). We extend their schematic to the version with delayed and noisy feedback with unknown delays and noise.

Let us consider the cost of selecting a new node at level $h > 0$ is $\lambda(Z_h)$ and the bias added in the decision due to the limited evaluation is $\zeta(Z_h)$. Here, Z_h is the internal state of the multi-fidelity evaluator which influences both the cost of evaluation and the bias in evaluation.

Thus, the multi-fidelity delayed-UCBV selection rule with unknown noise and delay becomes

$$B_{(h,i),S_{(h,i)}(t-1),t} \triangleq \hat{\mu}_{(h,i),S_{(h,i)}(t-1)} + \sqrt{\frac{2\hat{\sigma}^2_{(h,i),S_{(h,i)}(t-1)}\log t}{S_{(h,i)}(t-1)}} + \frac{3b\log t}{S_{(h,i)}(t-1)} + \zeta(Z_h). \quad (41)$$

Given this update rule and the multi-fidelity model, we observe that the Lemma 1 of (Sen et al., 2019) holds.

Lemma 2 (Lemma 1 (Sen et al., 2019))**.** *When the* PCTS *algorithm runs with a total budget* Λ *and the multi-fidelity selection rule (Equation* (41)*), then the total number of iterations that the algorithms runs for*

$$T(\Lambda) \geq H(\Lambda) + 1, \quad (42)$$

where

$$H(\Lambda) \triangleq \max\{H : \sum_{h=1}^{H} \lambda(Z_h) \leq \Lambda\}. \quad (43)$$

Given Lemma 2, we can retain the bounds of Theorem 2 and 4 by substituting $T = H(\Lambda)$.

Corollary 2 (Multi-fidelity PCTS with stochastic delays and noise)**.** *Let us consider that the expected objective function f satisfies Assumption 1, and its $4\nu_1/\nu_2$-near-optimality dimension is $d > 0$. The function evaluation has h-dependent fidelity such that $H(\Lambda) \triangleq \max\{H : \sum_{h=1}^{H} \lambda(Z_h) \leq \Lambda\}$ and the induced bias $\zeta(Z_h) = \nu_1\rho^h$.*

Then, under Assumption 2 and for any $d' > d$, PCTS *algorithm using* DUCB1 *achieves expected simple regret*

$$\epsilon_\Lambda = O\left((H(\Lambda))^{-\frac{1}{d'+2}}(\ln H(\Lambda) + \tau)^{\frac{1}{d'+2}}\right), \quad (44)$$

where the expected delay is upper bounded by τ, and PCTS *algorithm using* DUCBV *achieves expected simple regret*

$$\epsilon_\Lambda = O\left((H(\Lambda))^{-\frac{1}{d'+2}}(((\sigma/\nu_1)^2 + 2b/\nu_1)\ln H(\Lambda) + \tau)^{\frac{1}{d'+2}}\right), \quad (45)$$

where the maximum variance of the noise in evaluations is σ^2.

Proof. The proof relies on a simple observation by (Munos, 2014) and (Sen et al., 2019) that at each iteration of the HOO schematic only one point is selected randomly. Thus, we observe that the simple regret for PCTS with delayed-UCB1 is

$$\epsilon_\Lambda \leq \mathbb{E}\left[\mathbb{E}\left[\frac{\text{Reg}_T}{T} \mid T(\Lambda) = T\right]\right]$$

$$\leq \mathbb{E}\left[\left(\frac{\ln T}{T}\right)^{\frac{1}{d'+2}}\left(1+\frac{\tau}{\ln T}\right)^{\frac{1}{d'+2}} \mid T(\Lambda)=T\right]$$

In the first line, the inner expectation is over the randomness of a new node selection and the outer expectation is over the randomness of the computational budget due to inherent state of the multi-fidelity evaluator.

Now, we observe that the inner quantity in the last conditional expectation decreases with T and by Lemma 2, $T(\Lambda) \geq H(\Lambda)$ almost surely. Thus, we get

$$\epsilon_\Lambda = O\left((H(\Lambda))^{-\frac{1}{d'+2}}\left(\ln H(\Lambda)+\tau\right)^{\frac{1}{d'+2}}\right).$$

The proof for the PCTS with delayed-UCBV node selection proceeds similarly. □

B.4.1 Some Models of Cost Budget

Depending on the evaluation problem, we may have different cost functions. Here, we instantiate three such models of cost budget.

Let us denote the bias and cost functions of a multi-fidelity evaluators as $\zeta(.)$ and $\lambda(.)$, and the internal state of the evaluator is $Z_h = \zeta^{-1}(\nu_1\rho^h)$. Under this specification, we show four possible cost models:

1. **Linearly increasing cost:** $\lambda(Z_h) \leq \min\{\beta h, \lambda(1)\}$. This cost model is observed for hyperparameter tuning of deep neural networks.
   Under **cost model 1**,
   $$\Lambda \leq \sum_{h=1}^{H}\lambda(Z_h) \leq \sum_{h=1}^{H}\min\{\beta h,\lambda(1)\} \leq 1/2(\sum_{h=1}^{H}\beta h+\lambda(1)) \leq \frac{1}{4}(\beta H^2)+\lambda(1)/2).$$
   Thus,
   $$H(\Lambda) \geq \sqrt{2(2\Lambda-\lambda(1))/\beta}.$$
2. **Constant cost:** $\lambda(Z_h) \leq \min\{\beta, \lambda(1)\}$, where $\beta > 0$. This cost model is observed for hyperparameter tuning of deep neural networks.
   Under **cost model 2**,
   $$\Lambda \leq \sum_{h=1}^{H}\lambda(Z_h) \leq \sum_{h=1}^{H}\min\{\beta,\lambda(1)\} \leq \frac{\sum_{h=1}^{H}\beta+\lambda(1)}{2} \leq \frac{1}{2}(H\beta+\lambda(1)).$$
   Thus,
   $$H(\Lambda) \geq (2\Lambda-\lambda(1))/\beta.$$
3. **Polynomially decaying cost.** $\lambda(Z_h) \leq \min\{h^{-\beta}, \lambda(1)\}$, where $\beta > 0$ and $\beta \neq 1$. This cost model is observed for tuning database parameters in UDO.
   Under **cost model 3**,
   $$\Lambda \leq \sum_{h=1}^{H}\lambda(Z_h) \leq \sum_{h=1}^{H}\min\{h^{-\beta},\lambda(1)\}$$
   $$\leq \frac{\sum_{h=1}^{H}h^{-\beta}+\lambda(1)}{2} \leq \frac{1}{2}(\mathrm{Har}_\beta(H)+\lambda(1)) \leq \frac{1}{2}(\frac{H^{1-\beta}-1}{1-\beta}+\lambda(1)).$$
   Here, $\mathrm{Har}_\beta(H)$ is the generalised harmonic function of H of order β and is upper bounded by $\mathrm{RiemannZeta}(\beta)$. Thus,
   $$H(\Lambda) \geq (1+(1-\beta)(2\Lambda-\lambda(1)))^{1/(1-\beta)}.$$
4. **Exponentially decaying cost.** $\lambda(Z_h) \leq \min\{\beta^{-h}, \lambda(1)\}$, where $\beta \in (\rho, 1)$. This cost model is observed in tuning strongly convex functions with accelerated gradient descent.
   Under **cost model 4**,
   $$\Lambda \leq \sum_{h=1}^{H}\lambda(Z_h) \leq \sum_{h=1}^{H}\min\{\beta^{-h},\lambda(1)\} \leq \frac{\sum_{h=1}^{H}\beta^{-h}+\lambda(1)}{2} \leq \frac{1}{2}(\frac{\beta^{-H}-1}{1-\beta}+\lambda(1)).$$
   Thus,
   $$H(\Lambda) \geq \log_{1/\beta}\left(1+(1-\beta)(2\Lambda-\lambda(1))\right).$$

C Relaxing the Smoothness Assumptions

C.1 Local Smoothness with respect to the $\mathcal{T}$

Performing a global black-box optimization without any regularity assumption on the function is too good to be true. In tree search literature, initially, Agrawal (1995); Auer et al. (2007); Kleinberg et al. (2008b, 2019) have proposed to use global smoothness conditions on the objective functions. Bubeck et al. (2011) proposes to relax the pointwise global assumption over the whole domain $\mathcal{X}$ of f, for example global continuity or Lipschitzness (Goldstein, 1977), to the continuity condition around the optimum $f(x^*)$. In Section 2, we present the weak Lipschitzness assumption (Assumption 1) to elaborate the PCTS framework and later on to derive the corresponding regret bounds. We use this assumption to keep the results directly comparable with the original HOO algorithm (Bubeck et al., 2011). Now, we show that the assumption of weak Lipschitzness (Assumption 1) can be relaxed for PCTS to a completely *local* smoothness assumption (Assumption 4).

Grill et al. (2015) shows that the only smoothness condition required is local smoothness with respect to the partition constructed by the tree $\mathcal{T}$.

Assumption 4 (Local Smoothness with respect to the $\mathcal{T}$)**.** *Given the global maximizer $x^* \in \mathcal{X}$, we denote the index of the node at depth h containing x^* as l_h^*. Then, we assume that there exists a global maximizer x^* and smoothness parameters $\nu > 0$, $\rho \in (0, 1)$ such that,*

$$\forall h \geq 0, \forall x \in \mathcal{X}_{(h,l_h^*)}, \quad f(x) \geq f^* - \nu\rho^h. \tag{46}$$

This assumption shows that the only constrain we need on f is that along the optimal path of the covering tree. This is a plausible property in an optimization problem and also increases applicability of the algorithm significantly (Shang et al., 2019; Sen et al., 2018, 2019). The local assumption allows us to redefine the near-optimality dimension of Definition 1.

Definition 2 (Near-optimality Dimension with respect to the $\mathcal{T}$)**.** *For any $\nu > 0$, and $\rho \in (0, 1)$, the near-optimality dimension is defined as*

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

Here, $\mathcal{N}_h(\epsilon)$ is the number of nodes (h, l) such that $\sup_{x\in(h,l)} f(x) \geq f^* - \epsilon$. In other words, $\mathcal{N}_h(\epsilon)$ is the number of nodes at depth h that covers the ϵ-optimal region $\mathcal{X}_\epsilon$.

Given the corresponding smoothness assumptions, Definition 1 and 2 are analogous in intuition . They show the dependence of the global optimization problem on the volume of the near-optimal regions and their rate of shrinking with the depth of the tree (Auer et al., 2007). Specifically, $\mathcal{N}_h(2\nu\rho^h)$ is the number of nodes that any algorithm has to sample in order to find the optimum, and the optimization problem gets easier as the near-optimality dimension $d(\nu, \rho)$ decreases. Another interesting observation is that $d(\nu, \rho)$ depends only on f and $\mathcal{T}$, and not on the choice of the dissimilarity metric like Definition 1.

Given this new assumption, we restate the regret bound of Theorem 5.

Theorem 7. *Let us consider that the expected objective function f satisfies Assumption 1, and its $4\nu_1/\nu_2$-near-optimality dimension is $d > 0$. Then, under Assumption 2 and for any $d' > d$,* PCTS *algorithm uses a bandit algorithm MAB for node selection will achieve expected regret*

$$\mathbb{E}[\mathrm{Reg}_T] \leq 3\nu C(\nu, \rho) \sum_{h=0}^{H-1} \rho^{h(1-d(\nu,\rho))} + 3\nu\rho^H T + 6\nu C(\nu, \rho) \sum_{h=1}^{H} \rho^{(h-1)(1-d(\nu,\rho))} \times (U(T, \tau, \nu\rho^h)). \tag{47}$$

Here, $U(T, \tau, \nu\rho^h)$ is the upper bound on number of visits to the $2\nu\rho^h$-suboptimal nodes at depth $h > 0$ by BANDIT.

Proof. From Equation (17), we obtain the regret decomposition given three subtrees $\mathcal{T}_1$, $\mathcal{T}_2$, and $\mathcal{T}_3$.

$$\mathbb{E}[\mathrm{Reg}_T] = \mathbb{E}[\mathrm{Reg}_{T,1}] + \mathbb{E}[\mathrm{Reg}_{T,2}] + \mathbb{E}[\mathrm{Reg}_{T,3}] \tag{48}$$

Case 1: All nodes in I_h are $2\nu\rho^h$-optimal for all $h \geq 0$. Thus, all points in the corresponding subdomains are $3\nu\rho^H$-optimal (Assumption 4).

$$\mathbb{E}[\text{Reg}_{T,1}] \leq \sum_{h=0}^{H-1} 3\nu\rho^h |I_h| \leq \sum_{h=0}^{H-1} 3\nu\rho^h C(\nu,\rho)\rho^{-d(\nu,\rho)h} = 3\nu C(\nu,\rho) \sum_{h=0}^{H-1} \rho^{h(1-d(\nu,\rho))}. \quad (49)$$

The second inequality is a consequence of the Definition 2 of the near-optimality dimension $d(\nu,\rho)$.

Case 2: Since the nodes in $\mathcal{T}_2$ are $2\nu\rho^H$-optimal, all points in the corresponding subdomains are $3\nu\rho^H$-optimal (Assumption 4). Thus,

$$\mathbb{E}[\text{Reg}_{T,2}] \leq 3\nu_1\rho^H T. \quad (50)$$

Case 3: In $\mathcal{T}_3$, any node of J_h has a parent in I_{h-1}. Thus, the subdomains covered by nodes in J_h are at least $3\nu\rho^{h-1}$-optimal (Assumption 2 and 4). Thus, we obtain

$$\begin{aligned}
\mathbb{E}[\text{Reg}_{T,3}] &\leq \sum_{h=1}^{H} 3\nu\rho^{h-1} \sum_{l:(h,l)\in J_h} \mathbb{E}[S_{h,l}(T)] \\
&\leq \sum_{h=1}^{H} 3\nu\rho^{h-1} |J_h| \max_{(h,l)\in J_h} \mathbb{E}[S_{h,l}(T)] \quad (51) \\
&\leq \sum_{h=1}^{H} 6\nu\rho^{h-1} |I_{h-1}| \max_{(h,l)\in J_h} \mathbb{E}[S_{h,l}(T)] \quad (52) \\
&\leq \sum_{h=1}^{H} 6\nu\rho^{h-1} C(\nu,\rho)\rho^{-d(\nu,\rho)(h-1)} \max_{(h,l)\in J_h} \mathbb{E}[S_{h,l}(T)] \quad (53) \\
&\leq 6\nu C(\nu,\rho) \sum_{h=1}^{H} \rho^{(1-d(\nu,\rho))(h-1)} U(T,\tau,\nu\rho^h) \quad (54)
\end{aligned}$$

Equation (51) is obtained from the fact $\sum_{i=1}^{K} a_i \leq K \max_i a_i$. Equation (52) holds true as the parents of nodes of J_h are in I_{h-1}, and by the way the tree grows $|J_h| \leq 2|I_{h-1}|$ for any $h \geq 1$ (Branching factor = 2). Equation (53) is a direct consequence of the near-optimality dimension in Defintion 2. The last inequality is obtained as we denote the upper bound on number of visits to the $2\nu\rho^h$-suboptimal nodes at depth $h > 0$, i.e. nodes in J_h, by BANDITas $U(T,\tau,\nu\rho^h)$.

Combining Equations (49), (50), and (54) concludes the proof. □

If we consider $\nu_1 = \nu$, $d' = d(\nu,\rho)$, $C\nu_2^{-d'} = C(\nu,\rho)$, and $\delta(h) = \rho^h$, we obtain that the results of Theorem 5 and 7 only differ by constant factors. They also reduce the problem of bounding regret of PCTS to finding an upper bound on $U(T,\tau,\nu\rho^h)$ for a given BANDITalgorithm. Thus, we confirm that if we restate the regret bounds proved using Theorem 5 with the local smoothness assumption, they will have same dependency on T, τ, σ^2 and Λ, while d' changes to $d(\nu,\rho)$.

C.2 Unknown Smoothness

In Algorithm 1, we present a simplistic version of PCTS that takes the smoothness parameters (ν_1,ρ) as input. But in practice, we do not need to know the smoothness parameter. We take the Parallel Optimistic Optimizaton (POO) approach proposed by Grill et al. (2015) and later on extensively used in hierarchical tree search literature (Shang et al., 2019; Azar et al., 2014; Shang et al., 2018; Sen et al., 2018, 2019).

Given a hierarchical tree search algorithm $\mathcal{A}$, POO($\mathcal{A}$) takes maximum values of the smoothness parameters $(\nu_{1_{\max}},\rho_{\max})$ as input.[7] At first, POO($\mathcal{A}$) chooses $N = \frac{0.5\ln 2}{\ln(1/\rho_{\max})}\log(T/\log T)$ points $\{\rho_i\}_{i=1}^N$ in the interval $[0,\rho_{\max}]$, such that $\rho_i \triangleq \rho_{\max}^{\frac{2N}{i+1}}$. Then, it parallely spawns N instances of $\mathcal{A}$

[7] Both POO and $\mathcal{A}$ also take the branching factor of the tree, say K, as input. In PCTS, we fix K to 2. Thus, we omit mentioning it.

with smoothness parameters $(\nu_{1_{\max}}, \rho_i)$ as input. Finally, POO outputs the maximum of the optimal values computed by these N instances of $\mathcal{A}$. We denote the smoothness parameters corresponding to that tree as (ν^*, ρ^*). In brief, POO($\mathcal{A}$) performs a geometric line search for the parameter ρ^* in the interval $[0, \rho_{\max}]$ that maximizes the optimal values achieved by $\mathcal{A}(\nu_{1_{\max}}, \rho)$. Here, $\rho \in [0, \rho_{\max}]$.

Due to the multi-fidelity feedback, we adopt a specific version of POO, i.e. MFPOO (Sen et al., 2019, Algorithm 2), which is designed to be compatible with fixed budget and multi-fidelity feedback.

Theorem 8 (PCTS with Unknown Smoothness and DNF Feedback)**.** *If* PCTS *with DNF feedback is executed with parameters $\nu_{max}(\geq \nu^*)$, $\rho_{max}(\geq \rho^*)$, and a total cost budget Λ, then under Assumptions 4 and 2, the expected simple regret of at least one of the* PCTS + DUCB1 *instances is bounded by,*

$$\epsilon_\Lambda = O\left((\nu_{max}/\nu^*)^{D_{max}} \left(H(\Lambda/\log\Lambda)\right)^{-\frac{1}{d'+2}} \left(\tau + \ln H(\Lambda/\log\Lambda)\right)^{\frac{1}{d'+2}}\right). \tag{55}$$

and expected simple regret of at least one of the PCTS + DUCBV *instances is bounded by,*

$$\epsilon_\Lambda = O\left((\nu_{max}/\nu^*)^{D_{max}} \left(H(\Lambda/\log\Lambda)\right)^{-\frac{1}{d'+2}} \left(\tau + (\sigma^2 + 2b)\ln H(\Lambda/\log\Lambda)\right)^{\frac{1}{d'+2}}\right). \tag{56}$$

Here, $D_{max} = \log 2/\log(1/\rho_{max})$, σ^2 is the maximum noise variance, τ is the upper bound on expected delay, and b is the range of optimization domain.

Given the upper bound on the simple regret of the base algorithm $\mathcal{A}$ of $MFPOO(\mathcal{A})$ under multi-fidelity feedback, the proof technique of Theorem 2 in (Sen et al., 2019) can be reproduced. We merge Theorem 2 in (Sen et al., 2019) with the results of Theorem 2 to get the Theorem 8. Another interesting thing to note is that for MFPOO($\mathcal{A}$) to work Assumption 4 has to hold for one of the optimizers only, while it may spawn multiple maximizers (Grill et al., 2015; Shang et al., 2019).

D Additional Experimental Results

For the experimental analysis, we implement both PCTS + DUCB1σ *and* PCTS + DUCBV *with the local smoothness assumption and unknown smoothness parameters.*[8]

D.1 Details of Multi-fidelity Evaluations

We compare the performance of PCTS with: BO algorithms (BOCA (Kandasamy et al., 2017), GP-UCB (Srinivas et al., 2010), MF-GP-UCB (Kandasamy et al., 2016), GP-EI (Jones et al., 1998), MF-SKO (Huang et al., 2006)), tree search algorithms (MFPOO (Sen et al., 2018), MFPOO with UCB-V (Audibert et al., 2007)), zeroth-order GD algorithms (OGD, DBGD (Li et al., 2019)).[9]

In the section D.2, we show experiments of those algorithms on synthetic functions with DNF feedbacks. In the section D.3, we show the experiments for hyperparameter tuning of different machine learning models using those algorithms with DNF feedbacks. In the section D.4. we consider the stochastic delay instead of constant delay, and show experiment results for different algorithms under the stochastic delay.

D.2 Details of Optimizing Synthetic Functions

We evaluate the performance of aforementioned algorithm on those benchmark functions which are widely used in the black-box optimization literature (Sen et al., 2018, 2019). Those functions have been modified to incorporate the fidelity space $\mathcal{Z} = [0, 1]$ as (Sen et al., 2018, 2019) suggest.

D.2.1 Experiment Setup of Synthetic Functions

We run each experiment ten times for 600s on a MacBook Pro with a 6-core Intel(R) Xeon(R)@2.60GHz CPU and plot the median value of simple regret, shown in Figure 2. The delay time τ for all synthetic functions is set to four seconds. The noise is added from Gaussian distributions with corresponding variance σ^2. This σ is passed to UCB1-σ and DUCB1-σ in MFPOO and PCTS as it assumes the noise variance is known (Sen et al., 2019).For PCTS + DUCBV and MFPOO-UCBV, we do not pass the exact upper bound b of the function rather a loose upper bound on it, i.e. 5. For MFPOO and PCTS, we set the $\nu_{\max} = 0.95$ and $\rho_{\max} = 1.0$. The final optimal values of different algorithms on different synthetic functions are shown in Table 4. In all those experiments, either PCTS + DUCB11 or PCTS + DUCBV achieve the lowest simple regret.

D.2.2 Description of Synthetic Functions

Hartmann functions (van der Vlerk, 1996) We use two Hartmann functions in 3 and 6 dimensions, named as Hartmann3 and Hartmann6. The multi-fidelity object is

$$f_z(x) = \sum_{i=1}^{4} (\alpha_i - \alpha'(z)) \exp\left(-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2\right)$$

where $\alpha = [1.0, 1.2, 3.0, 3.2]$ and $\alpha'(z) = 0.1(1 - z)$.

For the hartmann3, the cost function is $\lambda(z) = 0.05 + (1 - 0.05)z^3$ and noise variance $\sigma^2 = 0.01$. The delay time τ is set to four seconds. The matrix A and P are,

$$A = \begin{bmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix} \qquad P = 10^{-4} \times \begin{bmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{bmatrix}$$

[8] Link to our code: `https://drive.google.com/drive/folders/188K6BoznXkdEWi8IOLclSNIktW_refR7`

[9] We use the implementations in `https://github.com/rajatsen91/MFTreeSearchCV` for baselines except OGD, DBGD, and MFPOO-UCBV.

For the hartmann6, the cost function is $\lambda(z) = 0.05 + (1 - 0.05)z^3$ and noise variance $\sigma^2 = 0.05$. The delay time τ is set to four seconds. The matrix A and P are,

$$A = \begin{bmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{bmatrix} \quad P = 10^{-4} \times \begin{bmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{bmatrix}$$

Currin exponential function (Currin et al., 1988). The input domain $\mathcal{X} = [0, 1]^2$. The cost function for CurrinExp is $\lambda(z) = 0.1 + z^2$ and the noise variance $\sigma^2 = 0.05$. The multi-fidelity object as a function of (x, z) is

$$f_z(x) = \left(1 - 0.1(1 - z) \exp\left(\frac{-1}{2x_2}\right)\right) \times \left(\frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20}\right)$$

Borehole function The cost function is $\lambda(z) = 0.1 + z^{1.5}$ and noise variance $\sigma^2 = 0.01$. The multi-fidelity object as a function of (x, z) is

$$f_z(x) = \frac{2z\pi T_u(H_u - H_l)}{\log(r/r_w)(1 + \frac{2LT_u}{\log(r/r_w)r_w^2 K_w} + \frac{T_u}{T_l})} + \frac{5(1 - z)T_u(H_u - H_l)}{\log(r/r_w)(1.5 + \frac{2LT_u}{\log(r/r_w)r_w^2 K_w} + \frac{T_u}{T_l})}$$

where $x = [r_w, r, T_u, H_u, T_l, H_l, L, H_w]$. The delay time τ is set to four seconds.

Branin function (Hey, 1979) The input domain $\mathcal{X} = [[-5, 10], [0, 15]]^2$. The function objective is

$$f_z(x) = a(x_2 - b(z)x_1^2 + c(z)x_1 - r)^2 + s(1 - t(z)) \cos(x_1) + s,$$

where $a = 1, b(z) = \frac{5.1}{4\pi^2} - 0.01(1 - z) c(z) = \frac{5}{\pi} - 0.1(1 - z), r = 6, s = 10$ and $t(z) = \frac{1}{8\pi} + 0.05(1 - z)$. When $z = 1$, it becomes the standard Branin function. The cost function is $\lambda(z) = 0.05 + z^3$ and noise variance $\sigma^2 = 0.05$. The delay time τ is set to four seconds.

D.2.3 Statistics of optimal values achieved by different optimizers

The median value of simple regret over ten runs are shown in Figure 2 with the error bar. The error bar indicates the spread of the maximum and minimum values obtained over the ten runs. The statistics of optimal values of different algorithms after 600s on different synthetic functions are shown in Table 4. In all the cases, we observe that either PCTS + DUCB1σ or PCTS + DUCBV outperforms the competing optimization algorithms.

Table 4: Maximum, Median, and Standard Deviation of optimal values over 10 runs of different algorithms for different synthetic functions.

Synthetic functions	Algorithms	Max value	Median value	Std. Dev.
Hartmann3 (optimal value = 3.86278)	MFPOO+UCB1-σ	3.862658	3.862658	4.71E-07
	MFPOO+UCBV	3.811498	3.811498	**6.84E-10**
	PCTS + DUCB1σ	3.862658	3.862658	4.70E-03
	PCTS + DUCBV	**3.862659688**	**3.8626584**	6.44E-07
	GP-UCB	3.8591	3.8591	8.21E-02
	MF-GP-UCB	3.8555	3.8555	2.97E-01
	BOCA	3.85496942	3.8474	7.03E-02
	GP-EI	3.8492	3.8492	8.59E-02
	MF-SKO	3.8511	3.8511	3.67E-02
	OGD	3.54411674	3.014917788	2.65E-01
	DBGD	3.54411674	3.54411674	2.78E-01
Hartmann6 (optimal value = 3.32237)	MFPOO+UCB1-σ	3.292949	3.287516333	4.38E-03
	MFPOO+UCBV	3.292949	2.958906875	2.55E-01
	PCTS + DUCB1σ	**3.306916432**	**3.305830186**	**1.54E-03**
	PCTS + DUCBV	**3.306916432**	3.298213481	7.11E-03
	GP-UCB	3.16314701	2.4833599	6.80E-01
	MF-GP-UCB	3.07564567	3.01829366	5.74E-02
	BOCA	3.08446115	2.94932738	1.35E-01
	GP-EI	2.58371424	2.551397575	3.23E-02
	MF-SKO	2.6617437	2.4059598	2.56E-01
	OGD	2.25086772	1.395420838	2.25E+00
	DBGD	2.954460562	1.916368356	1.35E+00
CurrinExp (optimal value = 13.798685)	MFPOO+UCB1-σ	13.697782	13.697675	6.48E-05
	MFPOO+UCBV	13.798306	13.798306	8.85E-05
	PCTS + DUCB1σ	**13.798685**	13.798396	1.36E-04
	PCTS + DUCBV	13.798585	**13.798585**	3.85E-02
	GP-UCB	**13.798685**	13.72859575	3.25E-02
	MF-GP-UCB	13.77983743	13.69857206	3.70E+00
	BOCA	13.798685	13.77240763	2.89E-02
	GP-EI	13.79599637	13.76940012	6.32E-02
	MF-SKO	13.67126373	13.67126373	1.86E-01
	OGD	13.73885226	13.73885226	**2.17E-11**
	DBGD	13.79819971	13.7891421	3.45E+00
Borehole (optimal value = 309.523221)	MFPOO+UCB1-σ	297.097139	294.6182492	1.91E+00
	MFPOO+UCBV	290.466079	285.769891	4.14E+00
	PCTS + DUCB1σ	305.9956224	305.4490009	**4.57E-01**
	PCTS + DUCBV	**308.3696046**	**305.8342653**	2.99E+00
	GP-UCB	288.8612949	284.6253047	4.09E+00
	MF-GP-UCB	278.6624033	274.664564	4.00E+00
	BOCA	278.0044636	269.1741653	9.29E+00
	GP-EI	285.1259936	283.1443876	2.26E+00
	MF-SKO	276.8185156	269.7879994	5.39E+00
	OGD	292.3678513	266.3082687	2.31E+01
	DBGD	296.0795646	231.7955644	4.55E+01
Branin (optimal value = -0.3979)	MFPOO+UCB1-σ	-0.519116	-0.52918	9.20E-03
	MFPOO+UCBV	-0.415716	-0.415716	**1.54E-03**
	PCTS + DUCB1σ	-0.4331555716	-0.4331555716	5.20E-02
	PCTS + DUCBV	**-0.3987907502**	**-0.3988127406**	8.58E-03
	GP-UCB	-0.41925899	-0.41925899	1.23E-01
	MF-GP-UCB	-3.24208941	-3.24208941	1.93E-02
	BOCA	-2.02091133	-2.02091133	2.86E-01
	GP-EI	-0.47900755	-0.47900755	9.94E-02
	MF-SKO	-1.1853606	-1.1853606	1.12E+00
	OGD	-9.524725088	-10.96088904	1.43E+01
	DBGD	-0.5061438122	-0.5205826043	1.02E+01

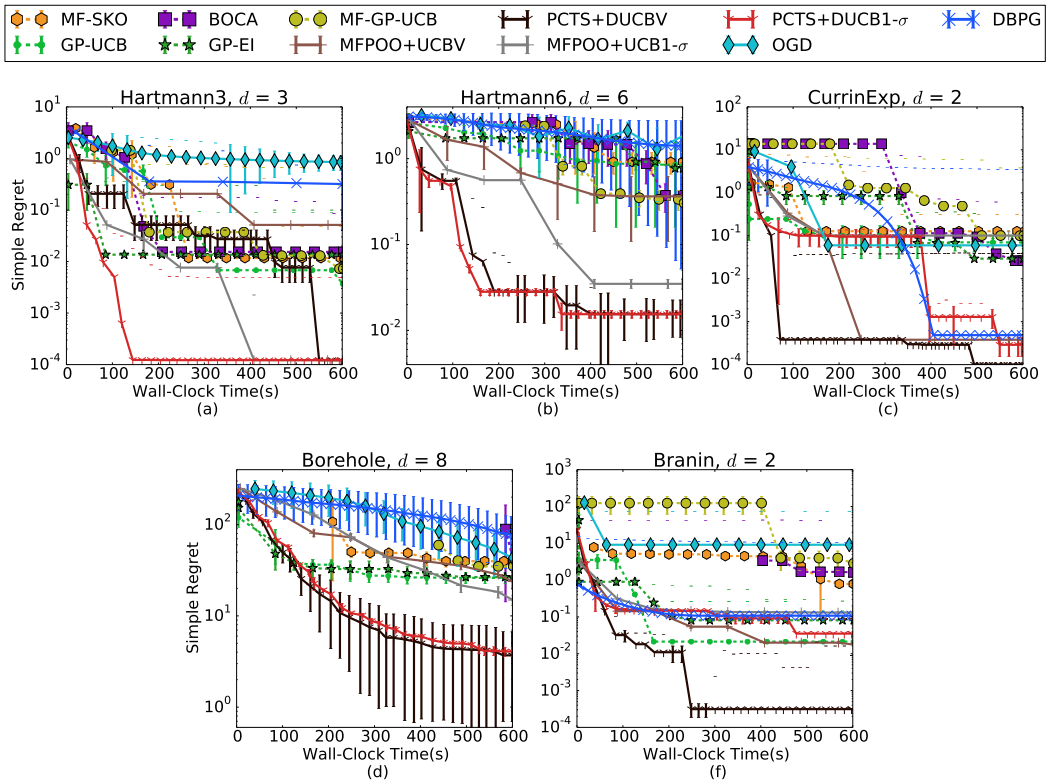

Figure 2: Figures (a) to (f) show simple regret (median of 10 runs) of different algorithms on synthetic functions with DNF feedbacks and the corresponding error bars $(y_{\text{median}} - y_{\min}, y_{\max} - y_{\text{median}})$ are plotted.

D.2.4 Key statistics of the trees constructed by MFPOO and PCTS

The detail comparisons between MFPOO and PCTS are shown in Table 5. In general, the depth of the tree created using PCTS algorithm is significantly larger than the depth of the tree created using MFPOO algorithm for both UCB1-σ and UCBV policy.

Table 5: Statistics of the trees constructed by MFPOO and PCTS based approaches for optimizing different synthetic functions.

Synthetic functions	Tree Search Algorithms	Tree Height	Number of Tree Nodes	Number of Iterations (T)	Best ρ	Best ν_1
Hartmann3	MFPOO+UCB1-σ	21	151	75	0.9259454628	0.006103201358
	MFPOO+UCBV	10	151	75	0.9259454628	0.006103201358
	PCTS + DUCB1σ	**27**	603	301	0.95	0.006103201358
	PCTS + DUCBV	18	681	340	0.95	0.006103201358
Hartmann6	MFPOO+UCB1-σ	25	151	75	0.9259454628	0.00842254613
	MFPOO+UCBV	25	151	75	0.9259454628	0.00842254613
	PCTS + DUCB1σ	36	593	296	0.95	0.00842254613
	PCTS + DUCBV	**39**	427	213	0.857375	0.01179156458
CurrinExp	MFPOO+UCB1-σ	15	151	75	0.9259454628	0.6094708386
	MFPOO+UCBV	12	151	75	0.9259454628	0.6094708386
	PCTS + DUCB1σ	**25**	607	303	0.9259454628	0.6094708386
	PCTS + DUCBV	18	915	457	0.95	0.6094708386
Borehole	MFPOO+UCB1-σ	42	151	75	0.9259454628	20.46960058
	MFPOO+UCBV	35	151	75	0.857375	28.65744081
	PCTS + DUCB1σ	**65**	597	298	0.857375	28.65744081
	PCTS + DUCBV	60	601	300	0.857375	28.65744081
Branin	MFPOO+UCB1-σ	20	151	75	0.9259454628	0.8313313704
	MFPOO+UCBV	13	151	75	0.9259454628	0.8313313704
	PCTS + DUCB1σ	**29**	599	299	0.95	0.8313313704
	PCTS + DUCBV	25	619	309	0.857375	1.163863919

Remark. Though we perform experiments for Gaussian noise, our analysis is valid for any noise with variance less than σ^2. In order to validate the claim, we also ran experiments with Laplace noise of variance $\sigma^2 = 0.05$ for Hartmann6, PCTS achieves an optimal value 3.2942 whereas the second one achieves 3.2562.

D.3 Optimizing Hyperparameters of Machine Learning Models

In this section, we show experiments about evaluating the aforementioned algorithms on a 32-core Intel(R) Xeon(R)@2.3 GHz server for hyperparameter tuning of real machine models. We tune hyper-parameters of SVM on News Group dataset, and XGB and Neural Network on MNIST datasets. The runtime of each experiment for those three tuning tasks are 700s, 1700s, and 1800s respectively. The median value of cross-validation accuracy is shown in Figure 3. The table 6 shows the final cross-validation accuracy found by different algorithms on those three tuning tasks. Notice that we invoke an additional simulation for all algorithms at fidelity $z = 1$ to obtain the cost for optimal fidelity and we preclude this initialization time for all algorithms. We set $\nu_{\max} = 1.0$, $\rho_{\max} = 0.95$ for MFPOO and PCTS algorithms. We set $\sigma^2 = 0.02$ for UCB1-σ used in PCTS + DUCB1σ and MFPOO-UCB1. For PCTS + DUCBV and MFPOO-UCBV, we use $b = 1$ as that is the maximum cross-validation accuracy achievable by any classification algorithm.

D.3.1 Description of Datasets and Models

News data on SVM. We train SVM classifier using different hyper-parameter tuning algorithms in NewsGroup dataset (Lang, 1995) The hyper-parameters to tune are the regularization term C, ranging from $[e^{-5}, e^{5}]$, and the kernel temperature γ from the range $[e^{-5}, e^{5}]$. Both are accessed in log scale. We set the delay τ to four seconds and the fidelity range $\mathcal{Z} = [0, 1]$ is mapped to $[100, 7000]$. The fidelity range represents the number of samples used to train the SVM classifier with the chosen parameters. The number of jobs specified for sklearn (Buitinck et al., 2013) is one. The 5-fold cross-validation accuracy is shown in Figure 3(a).

MNIST on XGB. We tune hyperparameters of XGBOOST (Chen and Guestrin, 2016) on the MNIST dataset (LeCun et al., 1998), where the hyperparameters are: (i) `max_depth` in $[2, 13]$, (ii) `n_estimators` in $[10, 400]$, (iii) `colsample_bytree` in $[0.2, 0.9]$, (iv) `gamma` in $[0, 0.7]$, and (v) `learning_rate` ranging from $[0.05, 0.3]$. The delay τ is set to ten seconds and the fidelity range $\mathcal{Z} = [0, 1]$ is mapped to the training sample range $[500, 20000]$. The number of jobs specified for sklearn (Buitinck et al., 2013) is three. The 3-fold cross-validation accuracy is shown in Figure 1(b).

MNIST on Deep Neural Network. We also apply the algorithms for tuning the hyper-parameters of a three layer multi layer perceptron (MLP) neural network (NN) classifier on the MNIST dataset (LeCun et al., 1998). Here, the hyper-parameters being tuned are: (i) number of neurons of the first, second, and third layers, which belong to the ranges $[32, 128]$, $[128, 256]$, and $[256, 512]$, respectively, (ii) initial learning rate of optimizer in $[e^{-1}, e^{-5}]$ (accessed in log-scale), (iii) optimizers from ('lbfgs', 'sgd', 'adam'), (iv) activation function from ('tanh', 'relu', 'logistic'), (v) early_stopping from ('enable', 'disable'). The delay τ for this experiment is 20 seconds. The number of training samples corresponding to the fidelities $z = 0$ and 1 are 1000 and 60000 respectively. The number of jobs specified for sklearn (Buitinck et al., 2013) is ten. The 3-fold cross-validation accuracy is shown in Figure 1(c).

D.3.2 Hyper-parameters found by PCTS + DUCBV

SVM on NewsGroup: {`kernel`: `rbf`, `C`: 5623.413251903499, `gamma`: 0.003162277660168379}

XGB on MNIST: {`n_estimators`: 302, `learning_rate`: 0.2375, `colsample_bytree`: 0.55, `gamma`: 0.35, `hidden_layer_sizes`: 4}

NN on MNIST:{`solver`: `adam`, `learning_rate_init`: 0.001, `learning_rate`: `invscaling`, `hidden_layer_sizes`: (104, 192, 320), `early_stopping`: False, `activation`: `relu`}

D.3.3 Hyper-parameters found by PCTS + DUCB1σ

SVM on NewsGroup: {`kernel`: `rbf`, `C`: 316.2277660168377, `gamma`: 0.003162277660168379}

XGB on MNIST: {`n_estimators`: 302, `learning_rate`: 0.175, `colsample_bytree`: 0.55, `gamma`: 0.35, `hidden_layer_sizes`: 10}

NN on MNIST: {`solver`: `adam`, `learning_rate_init`: 0.001, `learning_rate`: `invscaling`, `hidden_layer_sizes`: (104, 192, 448), `early_stopping`: False, `activation`: `relu`}

D.3.4 Statistics of optimal values achieved by different optimizers

The median value of 3-fold cross-validation accuracy over five runs are shown in Figure 3 with the error bar. The error bar indicates the spread of the maximum and minimum values obtained over the ten runs. The statistics of 3-fold cross-validation accuracy of three different classifiers tested on different real-world datasets after 700s, 1700s, and 1800s and over five runs are shown in Table 6. In all the cases, we observe that either PCTS + DUCB1σ or PCTS + DUCBV outperforms the competing optimization algorithms.

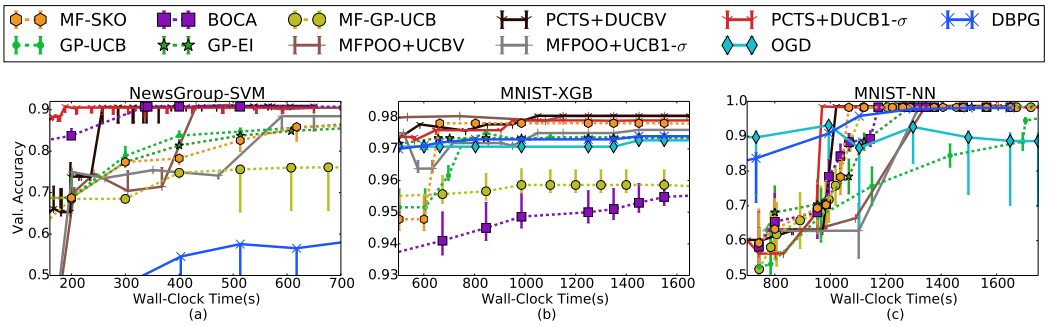

Figure 3: Figures (a) to (c) show the cross-validation accuracy (median of 5 runs) achieved on the hyperparameter tuning of classifiers on datasets with DNF feedbacks and the corresponding error bars $(y_{\text{median}} - y_{\min}, y_{\max} - y_{\text{median}})$ are plotted.

D.3.5 Key statistics of the trees constructed by MFPOO and PCTS

The detail comparisons between MFPOO and PCTS are shown in Table 7. In general, the depth of the tree created using PCTS algorithm is larger than the depth of the tree created using MFPOO algorithm for both UCB1-σ and UCBV policy.

Table 6: Maximum, median, and standard deviation of final cross-validation accuracy over 5 runs as achieved by different algorithms on different machine learning models tested on real datasets.

Tuning Task	Algorithms	Max value	Median value	Std
NewsGroup-SVM	MFPOO+UCB1-σ	0.888050296	0.8842852451	6.76E-02
	MFPOO+UCBV	0.9047538	0.9045731818	7.44E-03
	PCTS + DUCB1σ	0.9092364617	0.9081418236	2.37E-02
	PCTS + DUCBV	**0.910979**	**0.910269**	7.94E-03
	GP-UCB	0.8961291054	0.8821465296	1.75E-02
	MF-GP-UCB	0.7609988	0.76098258	2.09E-01
	BOCA	0.9034514049	0.9030594374	**6.53E-04**
	GP-EI	0.8774512931	0.871705938	1.15E-02
	MF-SKO	0.8792711233	0.858478645	6.93E-02
	OGD	0.06817823396	0.06542826139	3.93E-03
	DBGD	0.565711969	0.565711969	2.53E-01
MNIST-XGB	MFPOO+UCB1-σ	0.9783142575	0.9758995517	3.45E-03
	MFPOO+UCBV	0.98021339	0.9797497668	7.17E-03
	PCTS + DUCB1σ	**0.9819297976**	0.9791489477	3.97E-03
	PCTS + DUCBV	0.981874905	**0.9804613962**	2.36E-03
	GP-UCB	0.9748646302	0.9736495155	2.03E-03
	MF-GP-UCB	0.9583507463	0.9576005617	9.50E-03
	BOCA	0.9577324826	0.954150664	1.12E-02
	GP-EI	0.9775881491	0.9767505065	1.68E-03
	MF-SKO	0.9783595013	0.9780995824	6.50E-04
	OGD	0.9729382575	0.972700653	**4.75E-04**
	DBGD	0.9787723997	0.9781500036	2.07E-03
MNIST-NN	MFPOO+UCB1-σ	0.9845752279	0.9845167421	2.92E-02
	MFPOO+UCBV	0.9841946274	0.9840167256	2.97E-02
	PCTS + DUCB1σ	**0.9876736106**	**0.9872834306**	6.50E-02
	PCTS + DUCBV	0.9868071128	0.9865498949	4.29E-02
	GP-UCB	0.9446076809	0.9444498028	1.97E-02
	MF-GP-UCB	0.9832337681	0.9832332932	**1.58E-04**
	BOCA	0.9829225491	0.9828833355	9.80E-03
	GP-EI	0.9858101928	0.985799993	1.27E-03
	MF-SKO	0.9828856982	0.9828833355	5.91E-04
	OGD	0.9083502708	0.8981166245	4.67E-01
	DBGD	0.9813198067	0.9810665814	4.22E-02

Table 7: Statistics of the trees constructed by MFPOO and PCTS based approaches for hyper-parameter tuning of real machine learning models.

Tuning Task	Tree Search Algorithms	Tree Height	Number of Tree Nodes	Number of Iterations (T)	Best ρ	Best ν_1
NewsGroup-SVM	MFPOO+UCB1-σ	11	21	10	9.50E-01	5.49E-01
	MFPOO+UCBV	7	35	17	9.50E-01	5.49E-01
	PCTS + DUCB1σ	**21**	81	40	9.50E-01	4.49E-01
	PCTS + DUCBV	16	81	40	9.50E-01	3.88E-01
MNIST-XGB	MFPOO+UCB1-σ	4	9	4	8.15E-01	1.25E-01
	MFPOO+UCBV	4	9	4	8.57E-01	1.17E-01
	PCTS + DUCB1σ	8	17	8	9.03E-01	1.10E-01
	PCTS + DUCBV	**9**	17	8	9.03E-01	1.09E-01
MNIST-NN	MFPOO+UCB1-σ	5	9	4	9.03E-01	6.90E-01
	MFPOO+UCBV	5	11	5	9.03E-01	6.39E-01
	PCTS + DUCB1σ	**7**	15	7	9.50E-01	8.23E-01
	PCTS + DUCBV	**7**	15	7	9.50E-01	7.01E-01

D.4 Experiments with Stochastic Delays

In this section, we show experiment results for stochastic delays instead of constant delays. Same as previous experiments, We run each experiment ten times for 600s on a MacBook Pro with a 6-core Intel(R) Xeon(R)@2.60GHz CPU. The noise variance for all synthetic functions remains same as before and the initialization value of $\nu_{\max}$ and $\rho_{\max}$ for MFPOO and PCTS also keeps same.

Generating Stochastic Delays. Delays are generated using a geometric distribution $\tau \sim \text{Geo}(1/\bar{\tau})$, and the expectation of the delay time $\bar{\tau}$ for all synthetic functions is set to ten seconds. For generating stochastic delays using the geometric distribution, we use the same tricks and motivation as in (Vernade et al., 2017, Section 7).

D.4.1 Statistics of Optimal Values Achieved by Different Delay Tolerant Optimizers

The median value of simple regret over ten runs are shown in Figure 4 with the error bar. The error bar indicates the spread of the maximum and minimum values obtained over the ten runs. The statistics of the optimal values of different algorithms tested on different synthetic functions for 600s are shown in Table 8. In all the cases, we observe that either PCTS + DUCB1σ or PCTS + DUCBV outperforms the competing optimization algorithms.

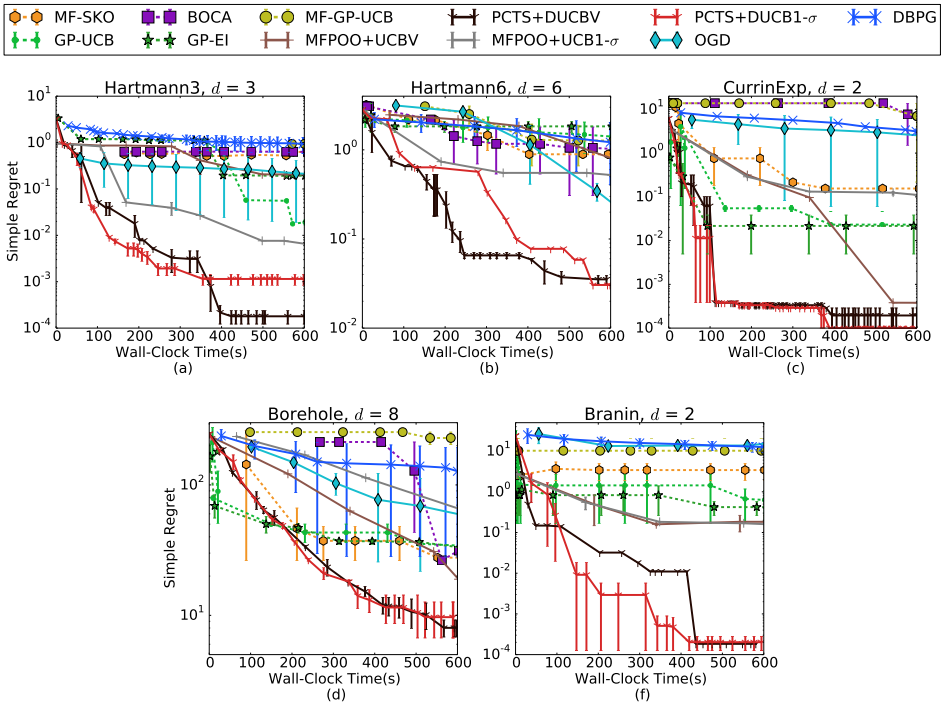

Figure 4: Figures (a) to (f) show simple regret (median of 10 runs) of different algorithms on synthetic functions under stochastic delay $\tau \sim \text{Geo}(0.1)$ and the corresponding error bars $(y_{\text{median}} - y_{\min}, y_{\max} - y_{\text{median}})$ are plotted.

Improvement of PCTS in constant and stochastic delays. Under constant delay, for Hartmann6 and Branin, PCTS achieve the optimal values as 3.305830186 and -0.3988127406 respectively, while the second best achieves the optimal value of f as 3.287516333 and -0.415716 respectively. Under stochastic delay, for Hartmann6 and Branin, PCTS has the optimal value of f as 3.291825 and -0.398084 respectively, while the second best has the optimal value of f as 3.147713 and -0.554698 respectively. *We observe that the performance improvement led by PCTS variants is even higher for stochastic delays than constant delays.* Similar conclusions can be drawn from other results.

Table 8: Maximum, Median, and Standard Deviation of optimal values over 10 runs of different algorithms for different synthetic functions under stochastic delay $\tau \sim \text{Geo}(0.1)$.

Synthetic functions	Algorithms	Max value	Median value	Std
Hartmann3 (optimal value = 3.86278)	MFPOO+UCB1-σ	3.858158	3.856659	5.90E-03
	MFPOO+UCBV	3.656709	3.656619	**2.99E-04**
	PCTS + DUCB1σ	3.862543	3.861633	9.55E-04
	PCTS + DUCBV	**3.862658**	**3.8626**	1.03E-03
	GP-UCB	3.846287	3.846113	6.28E-04
	MF-GP-UCB	2.927849	2.925782	2.09E-03
	BOCA	3.229559	3.229192	7.74E-04
	GP-EI	3.672403	3.66852	4.81E-03
	MF-SKO	3.229922	3.229182	1.24E-03
	OGD	3.844178	3.660379	1.84E-01
	DBGD	3.281405	2.922297	3.60E-01
Hartmann6 (optimal value = 3.32237)	MFPOO+UCB1-σ	2.801362	2.801252	9.92E-04
	MFPOO+UCBV	2.570945	2.570896	**9.95E-05**
	PCTS + DUCB1σ	**3.291905**	**3.291825**	5.41E-04
	PCTS + DUCBV	3.29112	3.287154	4.01E-03
	GP-UCB	2.551277	1.91283	6.39E-01
	MF-GP-UCB	2.661155	2.051153	6.11E-01
	BOCA	3.014768	2.312637	7.03E-01
	GP-EI	1.747662	1.488307	2.60E-01
	MF-SKO	2.524356	2.430526	9.41E-02
	OGD	3.251979	3.147713	1.05E-01
	DBGD	2.914959	2.104141	8.11E-01
CurrinExp (optimal value = 13.798685)	MFPOO+UCB1-σ	13.688384	13.688384	5.70E-04
	MFPOO+UCBV	13.798306	13.798306	4.95E-04
	PCTS + DUCB1σ	**13.798585**	**13.798491**	8.82E-04
	PCTS + DUCBV	**13.798585**	13.798398	**3.12E-04**
	GP-UCB	13.78547	13.77547	1.03E-02
	MF-GP-UCB	13.580857	6.790429	7.43E+00
	BOCA	12.450548	6.225274	7.22E+00
	GP-EI	13.793733	13.77704	1.73E-02
	MF-SKO	13.671398	13.643552	2.83E-02
	OGD	13.738852	11.310591	3.34E+00
	DBGD	11.091881	10.673992	4.21E-01
Borehole (optimal value = 309.523221)	MFPOO+UCB1-σ	276.52646	246.52646	3.02E+01
	MFPOO+UCBV	293.597767	290.597767	3.42E+00
	PCTS + DUCB1σ	**302.808514**	299.836555	3.52E+00
	PCTS + DUCBV	302.694521	**301.506202**	**1.80E+00**
	GP-UCB	277.294312	275.288875	3.00E+00
	MF-GP-UCB	105.501624	80.501624	3.38E+01
	BOCA	283.150628	278.150628	1.35E+01
	GP-EI	281.252433	276.426655	8.53E+00
	MF-SKO	283.150628	281.830216	2.25E+00
	OGD	297.752136	250.768674	5.65E+01
	DBGD	281.503929	182.224534	1.08E+02
Branin (optimal value = -0.3979)	MFPOO+UCB1-σ	-0.50178	-0.5814	8.04E-02
	MFPOO+UCBV	-0.554698	-0.554698	9.84E-04
	PCTS + DUCB1σ	**-0.398027**	-0.398102	1.03E-03
	PCTS + DUCBV	-0.398084	**-0.398084**	**1.34E-04**
	GP-UCB	-0.566961	-1.028222	4.61E-01
	MF-GP-UCB	-9.96478	-10.621964	6.58E-01
	BOCA	-19.24392808	-20.621964	1.04E+01
	GP-EI	-0.654828	-0.819048	1.65E-01
	MF-SKO	-2.315798	-3.820579	2.62E+00
	OGD	-15.050285	-16.180975	6.92E+00
	DBGD	-9.56788	-13.007573	4.02E+00

Table 9: Statistics of the trees constructed by MFPOO and PCTS based approaches for optimizing different synthetic functions under the stochastic delay $\tau \sim \text{Geo}(0.1)$.

Synthetic functions	Tree Search Algorithms	Tree Height	Number of Tree Nodes	Number of Iterations (T)	Best ρ	Best ν_1
Hartmann3	MFPOO+UCB1-σ	17	67	33	0.9259454628	0.006103201358
	MFPOO+UCBV	12	67	33	0.95	0.006103201358
	PCTS+DUCB1-σ	26	407	203	0.95	0.006103201358
	PCTS+DUCBV	**28**	339	169	0.95	0.006103201358
Hartmann6	MFPOO+UCB1-σ	16	69	34	0.9259454628	0.00842254613
	MFPOO+UCBV	9	63	31	0.857375	0.01179156458
	PCTS+DUCB1-σ	30	323	161	0.9259454628	0.01179156458
	PCTS+DUCBV	**33**	327	163	0.95	0.00842254613
CurrinExp	MFPOO+UCB1-σ	15	69	34	0.857375	0.6094708386
	MFPOO+UCBV	10	69	34	0.9259454628	0.6094708386
	PCTS+DUCB1-σ	19	407	203	0.95	0.6094708386
	PCTS+DUCBV	21	385	192	0.857375	0.853259174
Borehole	MFPOO+UCB1-σ	21	63	31	0.857375	28.65744081
	MFPOO+UCBV	34	69	34	0.9259454628	20.46960058
	PCTS+DUCB1-σ	53	399	199	0.857375	28.65744081
	PCTS+DUCBV	**73**	321	160	0.9259454628	20.46960058
Branin	MFPOO+UCB1-σ	18	67	33	0.95	0.8313313704
	MFPOO+UCBV	10	69	34	0.9259454628	0.8313313704
	PCTS+DUCB1-σ	**23**	389	194	0.9259454628	0.8313313704
	PCTS+DUCBV	22	385	192	0.9259454628	0.8313313704

D.4.2 Key statistics of the trees constructed by MFPOO and PCTS

The detail comparisons between MFPOO and PCTS under the stochastic delay $\tau \sim \text{Geo}(0.1)$ are shown in Table 9. In general, the depth of the tree created using PCTS algorithm is larger than the depth of the tree created using MFPOO algorithm for both UCB1-σ and UCBV policy.

D.5 Experimental comparison with Successive Rejections

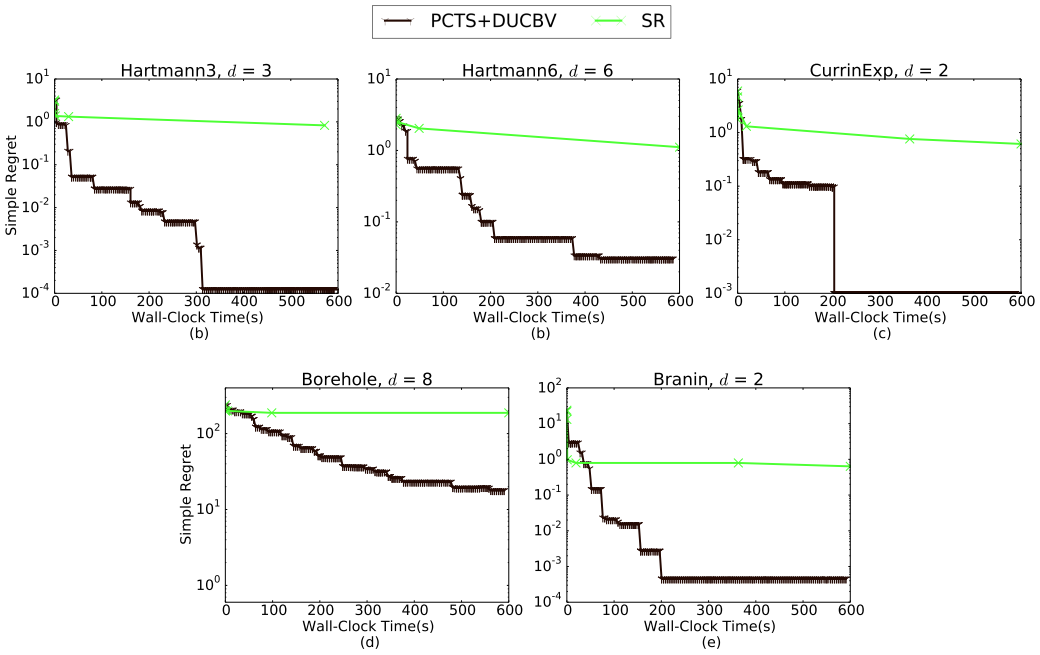

Figure 5: Figures (a) to (f) show simple regret (median of 5 runs) of PCTS + DUCBV and SR (successive rejection) on synthetic functions with delay feedbacks. We obtained almost same results for both PCTS + DUCBV and SR among those 5 experiments.

In order to compare with the wait-and-act version of the Successive Reject (SR) algorithm (Locatelli and Carpentier, 2018, Algorithm 2) with PCTS + DUCBV, we run experiments on five synthetic functions described in Appendix D.2. As the SR algorithm does not support multi-fidelity feedback, we set the feedbacks to have perfect fidelity. For comparison, also we set the delay τ to a constant, i.e. 4. The experimental results are shown in Figure 5. We observe for all the synthetic functions the simple regret of PCTS + DUCBV is at least 10X less than that of the modified SR algorithm handling constant delay. This shows that the modified SR algorithm enabled to handle constant delay does not yield lower simple regret than the proposed algorithm PCTS + DUCBV.