Gaussian Process Bandit Optimization with Few Batches

Zihan Li
National University of Singapore

Jonathan Scarlett
National University of Singapore

Abstract

In this paper, we consider the problem of black-box optimization using Gaussian Process (GP) bandit optimization with a small number of batches. Assuming the unknown function has a low norm in the Reproducing Kernel Hilbert Space (RKHS), we introduce a batch algorithm inspired by batched finite-arm bandit algorithms, and show that it achieves the cumulative regret upper bound $O^*(\sqrt{T\gamma_T})$ using $O(\log\log T)$ batches within time horizon T, where the $O^*(\cdot)$ notation hides dimension-independent logarithmic factors and γ_T is the maximum information gain associated with the kernel. This bound is near-optimal for several kernels of interest and improves on the typical $O^*(\sqrt{T}\gamma_T)$ bound, and our approach is arguably the simplest among algorithms attaining this improvement. In addition, in the case of a constant number of batches (not depending on T), we propose a modified version of our algorithm, and characterize how the regret is impacted by the number of batches, focusing on the squared exponential and Matérn kernels. The algorithmic upper bounds are shown to be nearly minimax optimal via analogous algorithm-independent lower bounds.

1 INTRODUCTION

Black-box optimization is the problem of maximizing an unknown function using only point queries that are typically expensive to evaluate. This problem has a wide range of applications including hyperparameter tuning (Snoek et al., 2012), experimental design (Srinivas et al., 2010), recommendation system (Vanchinathan et al., 2014), and robotics (Lizotte et al., 2007).

For example, tuning the hyperparameters of a machine learning model can be considered as a black-box optimization problem, where an evaluation of the function f at input x trains the model on hyperparameters x, and returns the cross-validation accuracy $f(x)$ on the validation set. Moreover, large models would make the evaluations expensive. Gaussian Process (GP) optimization is a prominent technique for maximizing black-box functions. The main idea is to place a prior over the unknown function and update the posterior distribution based on the noisy observations.

In this paper, we consider the problem of GP optimization in the case that observations are performed in batches, rather than being fully sequential. This is known to provide considerable benefits in settings where it is possible to perform multiple queries in parallel. The key distinction in our work compared to earlier works is that we consider the number of batches to be very low, namely, either independent of time horizon T, or growing very slowly as $O(\log\log T)$.

1.1 Related Work

Theoretical studies of GP optimization problems can be categorized into the Bayesian setting (i.e., the function is assumed to be drawn from a GP) or the non-Bayesian Reproducing Kernel Hilbert Space (RKHS) setting (i.e., the function has bounded RKHS norm). In this paper, we focus on the latter, which is often known as GP bandit optimization.

Classical GP optimization methods such as Gaussian process upper confidence bound (GP-UCB) (Srinivas et al., 2010), expected improvement (EI) (Mockus et al., 1978), and Thompson sampling (TS) (Thompson, 1933) iteratively choose points by optimizing their acquisition functions, and typically have a cumulative regret guarantee of $O^*(\sqrt{T}\gamma_T)$ under the RKHS setting, where the $O^*(\cdot)$ notation hides dimension-independent logarithmic factors, and γ_T is the maximum information gain associated with the kernel.

The $O^*(\sqrt{T}\gamma_T)$ scaling is not tight in general, and can even fail to be sublinear, including for the Matérn kernel with broad choices of the smoothness ν and

dimension d. The SupKernelUCB algorithm (Valko et al., 2013) overcomes this limitation and attains an $O^*(\sqrt{T\gamma_T})$ guarantee on discrete domains, which extends to continuous domain under mild Lipschitz-style assumptions. In view of recent improved bounds on γ_T (Vakili et al., 2021b), this nearly matches algorithm-independent lower bounds for the squared exponential (SE) and Matérn kernels (Scarlett et al., 2017).

SupKernelUCB is not considered to be practical (Calandriello et al., 2019), and a more practical algorithm guaranteeing sublinear regret for the Matérn kernel was given by Janz et al. (2020), albeit with suboptimal scaling. More recently, two more approaches were shown to attain $O^*(\sqrt{T\gamma_T})$ scaling: Robust Inverse Propensity Score (RIPS) for experimental design (Camilleri et al., 2021), and a tree-based domain-shrinking algorithm known as GP-ThreDS (Salgia et al., 2021). GP-ThreDS is shown to perform well empirically, while an advantage of the RIPS approach is that it requires at most $O(\log T)$ batches and is robust to model misspecification.

Several works have also proposed batch algorithms for GP optimization, including GP-UCB-PE (Contal et al., 2013), GP-BUCB (Desautels et al., 2014), and SynTS (Kandasamy et al., 2018). The focus in these works is on *fixed batch sizes*, and when the batch size is held fixed, the number of batches scales as $\Theta(T)$, in stark contrast to our focus. On the other hand, if a constant number of batches is used and all batches have equal size, then the first batch alone contributes $\Omega(T)$ regret. To overcome this, we will crucially rely on *variable-length* batches. While our algorithm and its analysis are new, some algorithms exist that also exploit varying batch lengths; for instance, the above-mentioned work of Camilleri et al. (2021) also falls in this category, as does the practical B3O algorithm that chooses points according to the peaks of an infinite mixture model (Nguyen et al., 2016).

While GP-BUCB enjoys a cumulative regret of $O^*(\sqrt{T\gamma_T})$ regardless of the batch size, this is only the case after having access to a suitably large offline initialization stage sampling T^{init} points that are not counted in the regret. If these points instead do contribute to the regret, then the bound becomes $O^*(T^{\text{init}} + \sqrt{T\gamma_T})$, which may be prohibitively large.

Our work builds on studies of batched algorithms for multi-armed bandit (MAB) problems (Cesa-Bianchi et al., 2013; Esfandiari et al., 2021). In particular, Cesa-Bianchi et al. (2013) attains a near-optimal bound using $O(\log \log T)$ batches, and Esfandiari et al. (2021) considers a constant number of batches. Our goal is to provide analogous results for the RKHS setting, and while we build on ideas from these MAB

works, the details are substantially different.

1.2 Contributions

We introduce a batch algorithm Batched Pure Exploration (BPE) for GP bandits, and show that it achieves the cumulative regret upper bound $O^*(\sqrt{T\gamma_T})$ within time horizon T using $O(\log \log T)$ batches. As outlined above, this is near-optimal for the SE and Matérn kernels. Compared to the alternatives SupKernelUCB, RIPS, and GP-ThreDS outlined above, we believe that our approach is the simplest (see Remark 1 below for discussion), and only relies on a basic GP-based confidence bound (Vakili et al., 2021a) without the need for the less elementary tools used previously.

In addition, in the case of a constant number of batches (i.e., not increasing with T), we propose a modified version of BPE, and characterize how the regret is impacted by the number of batches for the SE and Matérn kernels. Our bound is shown to be near-optimal by providing an analogous algorithm-independent lower bound.

2 PROBLEM SETUP

We consider the problem of optimizing a black-box function $f(x)$ over a compact domain $\mathcal{X} \subseteq \mathbb{R}^d$ with batched observations (possibly with varying batch lengths). The algorithm selects a single point x_t at time $t = 1, \ldots, T$, where the time horizon T is split into B batches indexed by $i = 1, \ldots, B$. The length of the i-th batch is denoted by N_i, and we use $t_i = \sum_{j=1}^{i} N_j$ to denote the time at the end of batch i. At the end of each batch, the algorithm observes the batch of noisy samples $y_t = f(x_t) + \epsilon_t$, where the noise terms ϵ_t are i.i.d. R-sub-Gaussian for all t.

We assume that $f \in \mathcal{F}_k(\Psi)$, where $\mathcal{F}_k(\Psi)$ denotes the set of all functions whose RKHS norm $\|f\|_k$ is upper bounded by some constant $\Psi > 0$, and we assume the kernel is normalized such that $k(x, x) \leq 1, \forall x \in \mathcal{X}$. We consider an algorithm that uses a fictitious[1] Bayesian GP model in which f is considered to be randomly drawn from a GP with prior mean $\mu_0(x) = 0$ and kernel function k. Given a sequence of points $(x_1, \ldots, x_t)$ and their noisy observations $\mathbf{y}_t = (y_1, \ldots, y_t)$ up to time t, the posterior distribution of f is a GP with mean and variance given by

$$\mu_t(x) = \mathbf{k}_t(x)^T(\mathbf{K}_t + \lambda\mathbf{I}_t)^{-1}\mathbf{y}_t \tag{1}$$

$$\sigma_t(x)^2 = k(x, x) - \mathbf{k}_t(x)^T(\mathbf{K}_t + \lambda\mathbf{I}_t)^{-1}\mathbf{k}_t(x), \tag{2}$$

---

[1] By "fictitious", we mean that the Bayesian model is only used by the algorithm, but the theoretical analysis itself is for the non-Bayesian RKHS model.

where $\mathbf{k}_t(x) = [k(x_i, x)]_{i=1}^t$, $\mathbf{K}_t = [k(x_t, x_{t'})]_{t,t'}$, and $\lambda > 0$ is a free parameter. Similarly, we denote the posterior mean and variance given only the points and observations in batch i by $\mu^i(x)$ and $\sigma^i(x)^2$.

We pay particular attention to two commonly-used kernels, namely, the squared exponential (SE) kernel and Matérn kernel:

$$k_{\text{SE}}(x, x') = \exp\left(-\frac{(d_{x,x'})^2}{2l^2}\right),$$

$$k_{\text{Matérn}}(x, x') = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}d_{x,x'}}{l}\right)^\nu B_\nu\left(\frac{\sqrt{2\nu}d_{x,x'}}{l}\right),$$

where $d_{x,x'} = \|x - x'\|$, $l > 0$ denotes the length-scale, ν is a smoothness parameter, and Γ and B_ν are the Gamma and Bessel functions. We denote the *maximum information gain* by (Srinivas et al., 2010)

$$\gamma_t = \max_{A \subseteq \mathcal{X}:|A|=t} I(f_A; y_A)$$
$$= \max_{x_1,\ldots,x_t} \frac{1}{2} \log \det(\mathbf{I}_t + \lambda^{-1}\mathbf{K}_t),$$

where $f_A = [f(x_t)]_{x_t \in A}$, $y_A = [y_t]_{x_t \in A}$, and $I(\cdot; \cdot)$ denotes mutual information (Cover and Thomas, 2006). γ_t quantifies the maximum reduction in uncertainty about f after t observations. We will use the following known upper bounds on γ_t from (Srinivas et al., 2010; Vakili et al., 2021b) in our analysis:

$$\gamma_t^{\text{SE}} = O^*((\log t)^d), \tag{3}$$

$$\gamma_t^{\text{Matérn}} = O^*\left(t^{\frac{d}{2\nu+d}}\right), \tag{4}$$

where the $O^*(\cdot)$ notation suppresses dimension-independent logarithmic factors.

To circumvent the $\sqrt{T}\gamma_T$ bottleneck discussed above, we avoid using standard confidence bounds (Srinivas et al., 2010; Chowdhury and Gopalan, 2017), and instead use a recent bound by Vakili et al. (2021a).[2] In view of these bounds only being valid for offline (i.e., non-adaptive) sampling strategies, we use an upper confidence bound (UCB) and lower confidence bound (LCB) defined using *only points within a single batch*:

$$\text{UCB}_i(x) = \mu^i(x) + \sqrt{\beta}\sigma^i(x), \tag{5}$$

$$\text{LCB}_i(x) = \mu^i(x) - \sqrt{\beta}\sigma^i(x), \tag{6}$$

where $i \in \{1, \ldots, B\}$ indexes the batch number, and β is an exploration parameter. The choice of β is dictated by the following result, which we emphasize only holds for points chosen in an offline manner (i.e., independent of the observations).

---

[2]Similar improved confidence bounds for non-adaptive sampling are also implicit in (Valko et al., 2013) and (Camilleri et al., 2021), but we find those of (Vakili et al., 2021a) more directly suitable for our purposes.

Algorithm 1 Batched Pure Exploration (BPE) for the case that the domain $\mathcal{X}$ is a finite set

Input: Domain $\mathcal{X}$, GP prior (μ_0, σ_0), time horizon T
1: $\mathcal{X}_1 \leftarrow \mathcal{X}, t \leftarrow 1, N_0 \leftarrow 1$
2: **for** $i \leftarrow 1, 2, \ldots$ **do**
3: $\quad N_i \leftarrow \lceil \sqrt{TN_{i-1}} \rceil$
4: $\quad \mathcal{S}_i \leftarrow \emptyset$
5: $\quad$ **for** $j \leftarrow 1, 2, \ldots, N_i$ **do**
6: $\quad\quad$ Compute σ_{j-1}^i only based on $\mathcal{S}_i$
7: $\quad\quad x_t \leftarrow \arg\max_{x \in \mathcal{X}_i} \sigma_{j-1}^i(x)$
8: $\quad\quad \mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{x_t\}$
9: $\quad\quad t \leftarrow t + 1$
10: $\quad\quad$ **if** $t > T$ **then**
11: $\quad\quad\quad$ Terminate
12: $\quad$ Observe all points in $\mathcal{S}_i$
13: $\quad$ Compute μ^i and σ^i only based on $\mathcal{S}_i$
14: $\quad \mathcal{X}_{i+1} \leftarrow \{x \in \mathcal{X}_i : \text{UCB}_i(x) \geq \max \text{LCB}_i\}$, where the confidence bounds use (5) and (6) with $\beta = \left(\Psi + \frac{R}{\sqrt{\lambda}}\sqrt{2\log\frac{|\mathcal{X}|B}{\delta}}\right)^2$ (here B is deterministic given T, and can be computed using line 3)

Lemma 1. (Vakili et al., 2021a, Theorem 1) *Let f be a function with RKHS norm at most Ψ, and let $\mu_t(x)$ and $\sigma_t(x)^2$ be the posterior mean and variance based on t points $\mathbf{X} = (x_1, \ldots, x_t)$ with observations $\mathbf{y} = (y_1, \ldots, y_t)$. Moreover, suppose that the t points in $\mathbf{X}$ are chosen independently of all samples in $\mathbf{y}$. Then, for any fixed $x \in \mathcal{X}$, it holds with probability at least $1 - \delta$ that $|f(x) - \mu_t(x)| \leq \sqrt{\beta(\delta)}\sigma_t(x)$, where $\beta(\delta) = \left(\Psi + \frac{R}{\sqrt{\lambda}}\sqrt{2\log\frac{1}{\delta}}\right)^2$.*

In the case of a finite domain, a union bound over all possible x immediately gives the following.

Corollary 1. *Under the setup of Lemma 1, if the domain $\mathcal{X}$ is finite, then for any $\delta \in (0, 1)$, it holds with probability at least $1 - \delta$ that $|f(x) - \mu_t(x)| \leq \sqrt{\beta(\frac{\delta}{|\mathcal{X}|})}\sigma_t(x)$ simultaneously for all $x \in \mathcal{X}$.*

To evaluate the algorithm performance, with $x^* = \arg\max_{x \in \mathcal{X}} f(x)$ being the optimal point, we let $r_t = f(x^*) - f(x_t)$ denote the simple regret incurred at time t, $R^i = \sum_{t=t_{i-1}+1}^{t_i} r_t$ denote the cumulative regret incurred in batch i, and $R_T = \sum_{i=1}^B R^i$ denote the overall cumulative regret incurred. Our objective is to attain small R_T with high probability.

Throughout the paper, we assume that the kernel parameters (ν, l) and dimension d are fixed constants (i.e., not depending on T).

3 SLOWLY GROWING NUMBER OF BATCHES

In this section, we study the scenario where we are allowed to choose the number of batches B as a function of T, and provide an algorithm and its theoretical regret upper bound. We select the batch sizes in a similar manner to the multi-armed bandit problem (Cesa-Bianchi et al., 2013) and suitably adapt the analysis. The resulting algorithm, BPE, is described in Algorithm 1; we initially focus on the case that $\mathcal{X}$ is finite, and later turn to continuous domains.

BPE maintains a set $\mathcal{X}_i$ of potential maximizers, and each batch consists of an exploration step and an elimination step to reduce the size of $\mathcal{X}_i$. In the exploration step, the algorithm explores the most uncertain regions in $\mathcal{X}_i$ by iteratively picking the point with the highest uncertainty, which is measured by the posterior variance. This is possible because the posterior variance can be computed using (2) without knowing any observed values. To ensure the validity of the offline assumption in Corollary 1, we ignore the previous batches and compute the posterior variance only based on the points selected in the current batch.[3] The number of points selected in the i-th batch is $N_i = \lceil \sqrt{T N_{i-1}} \rceil$, where we define $N_0 = 1$ for convenience. Note that these values of N_i can be computed offline (i.e., in advance), and thus, the total number of batches B is also deterministic given T.

In the elimination step, the algorithm eliminates points whose UCB is lower than the highest LCB from $\mathcal{X}_i$ (as is commonly done in MAB and GP bandit problems), as these points must be suboptimal (as long as the confidence bounds are valid). Hence, we only need to explore the updated $\mathcal{X}_i$ in next batch.

To re-iterate what we discussed previously, compared to regular GP optimization algorithms, our use of the offline confidence bound in Lemma 1 is the key to attaining a final dependence of $\sqrt{T\gamma_T}$ instead of $\sqrt{T}\gamma_T$.

The following proposition shows that the choice of batch size N_i in the BPE algorithm yields a number of batches given by $B = O(\log\log T)$.

Proposition 1. *With $N_i = \lceil \sqrt{T N_{i-1}} \rceil$ and $N_0 = 1$, the BPE algorithm terminates (i.e., reaches T selections) within at most $\lceil \log_2 \log_2 T \rceil + 1$ batches.*

Proof. Define $\widetilde{N}_i = \sqrt{T \widetilde{N}_{i-1}}$ with $\widetilde{N}_0 = 1$, which implies $\widetilde{N}_i = T^{\frac{1}{2}}(\widetilde{N}_{i-1})^{\frac{1}{2}} = T^{\frac{1}{2}+\frac{1}{4}}(\widetilde{N}_{i-2})^{\frac{1}{4}} = \cdots = T^{\frac{1}{2}+\frac{1}{4}+\cdots+\frac{1}{2^i}}(\widetilde{N}_0)^{\frac{1}{2^i}} = T^{1-2^{-i}}$. Let $\widetilde{B} = \lceil \log_2 \log_2 T \rceil$,

---

[3]The fact that the sub-domain $\mathcal{X}_i$ depends on the samples from previous batches is inconsequential, because we do not use those samples in the new posterior calculation.

which implies $\widetilde{N}_{\widetilde{B}} = T/T^{2^{-\widetilde{B}}} \geq T/T^{\frac{1}{\log_2 T}} = T/2$. Since $\sum_{i=1}^{\widetilde{B}+1} N_i \geq N_{\widetilde{B}} + N_{\widetilde{B}+1} \geq \widetilde{N}_{\widetilde{B}} + \widetilde{N}_{\widetilde{B}+1} \geq T$, we must have $B \leq \widetilde{B} + 1$. $\qquad\square$

Our first main result is stated as follows.

Theorem 1. *Under the setup of Section 2 with a finite domain $\mathcal{X}$, choosing $\lambda = R^2$, the BPE algorithm yields with probability at least $1 - \delta$ that*

$$R_T = O^*(\Lambda \sqrt{T\gamma_T}),$$

where $\Lambda = \Psi + \sqrt{\log \frac{|\mathcal{X}|}{\delta}}$. In particular,

- *for the SE kernel, $R_T = O^*(\Lambda \sqrt{T(\log T)^d})$;*

- *for the Matérn kernel, $R_T = O^*(\Lambda T^{\frac{\nu+d}{2\nu+d}})$.*

The proof is given in Appendix A.1, and is based on the fact that after performing maximum-uncertainty sampling for N rounds, we are guaranteed to uniformly shrink the uncertainty to $O\left(\sqrt{\frac{\gamma_N}{N}}\right)$. From this result, our choices of batch lengths turn out to provide a per-batch cumulative regret of $O(\sqrt{T\gamma_T})$, which gives the desired result.

Remark 1. As discussed previously, we have provided a new approach to attaining the improved $O^*(\sqrt{T\gamma_T})$ dependence, and we are the first to do so using $O(\log\log T)$ batches, improving on the $O(\log T)$ batches used by Camilleri et al. (2021). Perhaps more importantly, we believe that the simplicity of our algorithm and analysis are particularly desirable compared to the existing works attaining $O^*(\sqrt{T\gamma_T})$ scaling (Valko et al., 2013; Salgia et al., 2021; Camilleri et al., 2021). Briefly, some less elementary techniques used in these works include adaptively partitioning the time indices into carefully-chosen subsets of $\{1, \ldots, T\}$ (Valko et al., 2013), tree-based adaptive partitioning of $\mathcal{X}$ (Salgia et al., 2021), and experimental design over the $|\mathcal{X}|$-dimensional simplex by Camilleri et al. (2021).

3.1 Extension to Continuous Domains

While we have focused on discrete domains for simplicity, extending to continuous domains is straightforward. For concreteness, here we discuss the case that $\mathcal{X} = [0, 1]^d$.

As discussed by Janz et al. (2020), the simplest approach to establishing analogous regret guarantees in this case is to construct a finite subdomain $\widetilde{\mathcal{X}}$ of $T^{d/2}$ points, with equal spacing of width $\frac{1}{\sqrt{T}}$ in each dimension. If we apply Algorithm 1 on the finite domain $\widetilde{\mathcal{X}}$, then the regret guarantee in Theorem 1 immediately holds with respect to the restricted-domain function,

with $\Lambda = \Psi + \sqrt{\frac{d}{2}\log T + \log\frac{1}{\delta}}$. To convert such a guarantee to the entire continuous function, we can use the following lemma.

Lemma 2. (Shekhar and Javidi, 2020, Proposition 1 and Remark 5) *For $f \in \mathcal{F}_k(\Psi)$ on a compact continuous domain, if k is the SE or Matérn kernel (with $\nu > 1$) and Ψ is constant, then f is guaranteed to be Lipschitz continuous with some constant L depending only on the kernel parameters.*

This Lipschitz continuity property establishes that $f(x^*) = \max_{x\in\mathcal{X}} f(x) \le \max_{\widetilde{x}\in\widetilde{\mathcal{X}}} f(\widetilde{x}) + O\left(\frac{L}{\sqrt{T}}\right)$, which implies that to convert the discrete-domain cumulative regret to the continuous domain, we only need to add $O(L\sqrt{T})$ to the upper bound in Theorem 1, which has no impact on the scaling laws for fixed L.

A more sophisticated approach to handling continuous domains is given by Vakili et al. (2021a), based on forming a fine discretization *purely for the theoretical analysis* and not for use by the algorithm itself. Such an approach can also be adopted here, and leads to a regret bound with the same scaling as the direct approach outlined above.[4] The details of this approach are somewhat more technical than the basic approach above. In particular, as observed in (Vakili et al., 2021a), we need to not only establish that the discretization has a minimal impact on f, but also on the posterior mean. A slight difference compared to (Vakili et al., 2021a) is that we additionally need to consider the impact of the discretization on the posterior standard deviation, since it is used in the elimination step of our algorithm. The full details be found in a recent paper that builds on our work while incorporating sparse GP approximations (Vakili et al., 2022); the analysis therein equally applies to the case that an exact GP model is used.

4 CONSTANT NUMBER OF BATCHES

While $B = O(\log\log T)$ is rather small, it is also of significant interest to understand scenarios where the number of batches is even further constrained. In this section, we consider the case that B is a pre-specified constant (not depending on T).

Throughout the section, we focus our attention on al-

gorithms for which the batch lengths are pre-specified, i.e., the algorithm *cannot* vary the batch lengths based on previous observations. For the algorithmic upper bounds such a property is clearly a strength, whereas for the algorithm-independent lower bounds, this leaves open the possibility of adaptively-chosen batch sizes further improving the performance. We discuss the challenges regarding lower bounds for adaptively-chosen batch lengths in Appendix B.

4.1 Upper Bound

We adapt BPE by carefully modifying the batch sizes, and in contrast to Theorem 1, we do this in a kernel-dependent manner. The resulting bounds for the SE and Matérn kernels are stated as follows.

Theorem 2. *Under the setup of Section 2 with an arbitrary constant number of batches $B \ge 2$,*

- *for the SE kernel, let $N_i = \left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta^i}{1-\eta^B}}(\log T)^d \right\rceil$, where $\eta = \frac{1}{2}$;*

- *for the Matérn kernel, let $N_i = \left\lceil T^{\frac{1-\eta^i}{1-\eta^B}} \right\rceil$, where $\eta = \frac{\nu}{2\nu+d}$;*

for $i = 1, \ldots, B-1$, and $N_B = T - \sum_{j=1}^{B-1} N_j$. Then, choosing $\lambda = R^2$, the corresponding modified BPE algorithm yields with probability at least $1 - \delta$ that

- *for the SE kernel, it holds that $R_T = O^*\left(\Lambda T^{\frac{1-\eta}{1-\eta^B}}(\log T)^{\frac{\eta-\eta^B}{1-\eta^B}d}\right) = O^*\left(\Lambda T^{\frac{2^B-2^{B-1}}{2^B-1}}(\log T)^{\frac{2^{B-1}-1}{2^B-1}d}\right);$*

- *for the Matérn kernel, $R_T = O^*\left(\Lambda T^{\frac{1-\eta}{1-\eta^B}}\right);$*

where $\Lambda = \Psi + \sqrt{\log(|\mathcal{X}|B/\delta)}$.

The proof is given in Appendix A.2, and is again based on the fact that N rounds of uncertainty sampling reduces the maximum uncertainty to $O\left(\sqrt{\frac{\gamma_N}{N}}\right)$. In this case, unlike Theorem 1, the batch lengths $\{N_i\}_{i=1}^B$ are specifically chosen to ensure that roughly the same (upper bound on) regret is incurred in every batch.

We observe that in both cases, the leading term in the upper bound is $T^{\frac{1-\eta}{1-\eta^B}}$, which decreases as B increases. The limiting leading term of $T^{1-\eta}$ as $B \to \infty$ is also consistent with Theorem 1.

Using the argument given in Section 3.1, the regret bound in Theorem 2 again extends directly to continuous domains.

---

[4]This is under the assumption that the algorithm can exactly optimize its acquisition function over the continuous domain, and over any sub-domain formed as a result of action elimination. While this is typically not to be expected in practice, our analysis also goes through essentially unchanged when the algorithm is only required to find a point whose posterior variance is within a constant factor (e.g., $\frac{1}{2}$) of the maximum.

4.2 Lower Bound

Since our focus is on the dependence on T and B, we assume in the following that the RKHS norm bound Ψ is a fixed constant. Recall also that the kernel parameters (ν, l) and dimension d are considered to be fixed constants. Throughout this subsection, we consider the standard rectangular domain $\mathcal{X} = [0,1]^d$ in order to utilize existing lower bounds that were proved for this domain.

While our focus is on the cumulative regret, an existing lower bound on the simple regret (for general algorithms without constraints on the number of batches) turns out to serve as a useful stepping stone. The simple regret is defined as $r^T = f(x^*) - f(x^T)$, where x^T is an additional point returned after T rounds. Then, we have the following.

Lemma 3. (Cai and Scarlett, 2021, Theorem 1) *Fix $\epsilon \in (0, \frac{1}{2})$, $\Psi > 0$, and $T \in \mathbb{Z}$. Suppose there exists an algorithm that, for any $f \in \mathcal{F}_k(\Psi)$, achieves average simple regret $\mathbb{E}[r^T] \leq \epsilon$ with probability at least $\frac{3}{4}$.[5] Then, if $\frac{\epsilon}{\Psi}$ is sufficiently small, we have*

1. *for the SE kernel, it is necessary that*

$$T = \Omega\left(\frac{1}{\epsilon^2} \left(\log \frac{1}{\epsilon} \right)^{d/2} \right),$$

2. *for the Matérn kernel, it is necessary that*

$$T = \Omega\left(\frac{1}{\epsilon^{2+d/\nu}} \right).$$

The following corollary provides a lower bound on the cumulative regret in the m-th batch, R^m, given an upper bound on t_{m-1} and a lower bound on t_m, where we recall that t_i is final time index of the i-th batch.

Corollary 2. *Fix constants a and b such that $0 < a < b \leq 1$, as well as a batch index m. Consider any batch algorithm whose batch sizes are fixed in advance, and satisfy*

1. *for the SE kernel, $t_{m-1} < \left(\frac{T}{(\log T)^{d/2}} \right)^a (\log T)^{d/2}$ and $t_m \geq \left(\frac{T}{(\log T)^{d/2}} \right)^b (\log T)^{d/2}$;*

2. *for the Matérn kernel, $t_{m-1} < T^a$ and $t_m \geq T^b$.*

Then, there exists $f \in \mathcal{F}_k(\Psi)$ such that with probability at least $\frac{3}{4}$, the cumulative regret incurred in batch m is upper bounded as follows:

1. *for the SE kernel, $R^m = \Omega\left(\left(\frac{T}{(\log T)^{d/2}} \right)^{b-a/2} (\log T)^{d/2} \right)$;*

2. *for the Matérn kernel; $R^m = \Omega\left(T^{b - \frac{\nu}{2\nu+d}a} \right)$.*

Proof. For the SE kernel, the inequalities $t_{m-1} < \left(\frac{T}{(\log T)^{d/2}} \right)^a (\log T)^{d/2}$ and $t_m \geq \left(\frac{T}{(\log T)^{d/2}} \right)^b (\log T)^{d/2}$ (with $b > a$) indicate that the size of the m-th batch satisfies $N_m \geq C_m \left(\frac{T}{(\log T)^{d/2}} \right)^b (\log T)^{d/2}$ for some constant C_m. Now suppose on the contrary that there exists an algorithm with t_{m-1} and t_m bounded as above such that, for any $f \in \mathcal{F}_k(\Psi)$, we have $R^m = o\left(\left(\frac{T}{(\log T)^{d/2}} \right)^{b-a/2} (\log T)^{d/2} \right)$ with probability at least $\frac{3}{4}$. Then, the average regret of points in the m-th batch is $\frac{R_m}{N_m} = o\left(\left(\frac{T}{(\log T)^{d/2}} \right)^{-a/2} \right)$ due to the lower bound on N_m. Substituting $\epsilon = o\left(\left(\frac{T}{(\log T)^{d/2}} \right)^{-a/2} \right)$ into Lemma 3,[6] we find that this algorithm must have $t_{m-1} = \sum_{i=1}^{m-1} N_i = \omega\left(\left(\frac{T}{(\log T)^{d/2}} \right)^a (\log T)^{d/2} \right)$. However, this contradicts our assumption of $t_{m-1} < \left(\frac{T}{(\log T)^{d/2}} \right)^a (\log T)^{d/2}$. Hence, the claimed lower bound on R_T must hold.

Similarly, for the Matérn kernel, the inequalities $t_{m-1} < T^a$ and $t_m \geq T^b$ indicate the size of the m-th batch satisfies $N_m \geq C_m T^b$ for some constant C_m. Now suppose on the contrary that there exists an algorithm with t_{m-1} and t_m bounded as above such that, for any $f \in \mathcal{F}_k(\Psi)$, we have $R^m = o\left(T^{b - \frac{\nu}{2\nu+d}a} \right)$ with probability at least $\frac{3}{4}$. Then, the average simple regret of the m-th batch is $\frac{R_m}{N_m} = o\left(T^{-\frac{\nu}{2\nu+d}a} \right)$ due to the lower bound on N_m. Substituting $\epsilon = o\left(T^{-\frac{\nu}{2\nu+d}a} \right)$ into Lemma 3, we find that this algorithm must have $t_{m-1} = \sum_{i=1}^{m-1} N_i = \omega(T^a)$. However, this contradicts our assumption of $t_{m-1} < T^a$. Hence, the claimed lower bound on R_T must hold. $\square$

Using Corollary 2 as a building block, we can obtain our main lower bound, stated as follows.

Theorem 3. *For any batch algorithm with a constant number B of batches and batch sizes fixed in advance, there exists $f \in \mathcal{F}_k(\Psi)$ such that we have with probability at least $\frac{3}{4}$ that*

1. *for the SE kernel, $R_T = \Omega\left(T^{\frac{1-\eta}{1-\eta^B}} (\log T)^{\frac{d(\eta - \eta^B)}{2(1-\eta^B)}} \right)$, where $\eta = \frac{1}{2}$;*

2. *for the Matérn kernel, $R_T = \Omega\left(T^{\frac{1-\eta}{1-\eta^B}} \right)$, where $\eta = \frac{\nu}{2\nu+d}$.*

The proof is given in Appendix A.3, with the main idea being that Corollary 2 characterizes how the re-

---

[5]The value $\frac{3}{4}$ can be replaced by any fixed constant in $(\frac{2}{3}, 1)$, and doing so does not affect the scaling laws.

[6]When applying Lemma 3, we substitute the (single) batch length for T, and let x^T be chosen uniformly at random from the set of sampled points in the batch.

gret must scale in each batch depending on the pre-specified batch lengths. We show that no matter how the batch lengths are chosen, there will always be at least one "bad batch" (i.e., the batch is too long and/or there were not enough prior samples to reduce the uncertainty a sufficient amount) that incurs the specified lower bound on regret.

Theorem 3, together with Theorem 2 (with Ψ treated as a constant), establishes that the overall cumulative regret for the modified BPE algorithm under the constant B setting is $R_T = \Theta^*(T^{\frac{1-\eta}{1-\eta^B}}(\log T)^{\Theta(d)})$ for the SE kernel and $R_T = \Theta^*\big(T^{\frac{1-\eta}{1-\eta^B}}\big)$ for the Matérn kernel. Thus, we have established matching behavior in the upper and lower bounds.

5 EXPERIMENTS

While the goals of this paper are theoretical in nature, here we support our findings with some basic proof-of-concept experiments. We emphasize that we do not attempt to claim state-of-the-art practical performance against other batch algorithms.

We produce two synthetic 2D functions and execute our algorithm with different values of B. We fix $|\mathcal{X}| = 2500$ points by evenly splitting the domain into a 50×50 grid. We let $T = 1000$ and $\sigma = 0.02$ and use the SE kernel with $l = 0.5$ as prior. For each function, we consider five optimization algorithms:

(1) Orig-BPE: Original BPE with batch sizes as stated in Algorithm 1.

(2-4) $\{3, 4, 6\}$-BPE: modified BPE with $B = 3, 4, 6$ respectively and batch sizes as stated in Theorem 2 but with the logarithmic term ignored. As a slight practical variation, after forming the initial batch lengths $N_1, \ldots, N_B$, we normalize them by $\frac{T}{\sum_{i=1}^{B} N_i}$ to maintain an overall length of T.

(5) GP-UCB, as a representative non-batch algorithm (Srinivas et al., 2010).

We use f_1 shown in Figure 1, a function with one obvious maximum, and f_2 shown in Figure 2, a function with multiple near-optimal local maxima. We produce our results by performing 10 trials and plotting the average cumulative regret, with error bars indicating half a standard deviation. As is common in the literature, instead of relying on the theoretical choice of β, we set it based on a minimal amount of tuning, choosing $\beta = 2$ for f_1 (less exploration needed to find the clear peak) and $\beta = 6$ for f_2 (encouraging slightly more exploration).
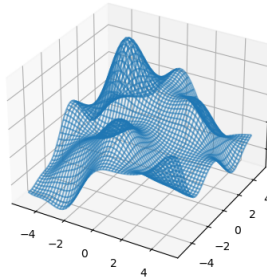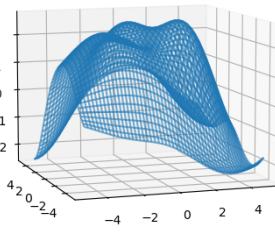

Figure 1: f_1 Figure 2: f_2

Comparison of algorithms. We compare the algorithms for f_1 and f_2 in Figures 3 and 4, respectively. While the curves for $B \in \{3, 4, 6\}$ are often close, there is a clear trend that smaller B tends to incur higher regret, which is consistent with our theory. We also observe that 6-BPE can outperform Orig-BPE, which is natural because the latter only uses 4 batches when $T = 1000$ (the growth of $\log \log T$ is extremely slow). GP-UCB unsurprisingly has the smallest regret, as it has the benefit of being fully sequential, though the large error bars in Figure 3 indicate that compared to BPE, it can suffer more from "failure runs" when β is chosen too aggressively (small).

Fixed vs. varying batch lengths. In Figures 5 and 6, we compare our varying batch lengths to fixed batch lengths. We observe that fixed batch lengths tend to be highly suboptimal, though the gap shrinks as B increases. This supports the fact that varying batch lengths are essential for obtaining near-optimal regret guarantees, as discussed in the introduction.

Partial vs. full posterior. Recall that our theoretical analysis crucially relied on ignoring previous batches in order to apply Lemma 1. Next, we investigate whether using all sampled points to form a "full posterior" can also be effective in practice. In this case, prior work suggests that β should logarithmically increase with the batch index, so along with the above choices of $\beta = 2$ and $\beta = 6$, we consider the use of $\beta_i = 3 \log(2i)$ in the i-th batch, which is analogous to the choices made in existing works (e.g., (Srinivas et al., 2010; Rolland et al., 2018)). The results are shown in Figures 7 and 8 (fixed β) and Figures 9 and 10 (growing β_i).

The curves in Figures 7 and 8 generally reach similar regret values as Figures 3 and 4, suggesting that using the full posterior can also be effective, though establishing the corresponding theory appears to be difficult. Another viewpoint on this finding is that although discarding previously-collected data may seem wasteful, it does not appear to harm the performance here. In these figures, we also observe an effect of the
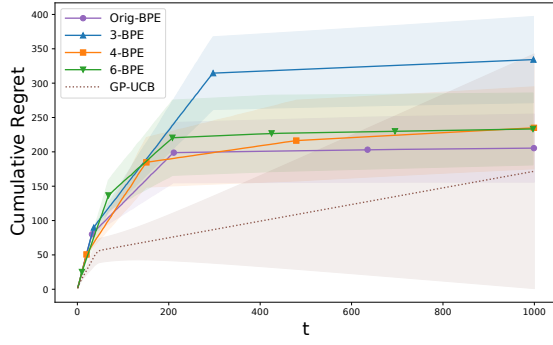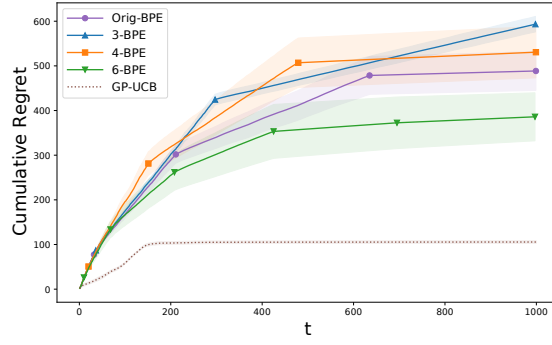
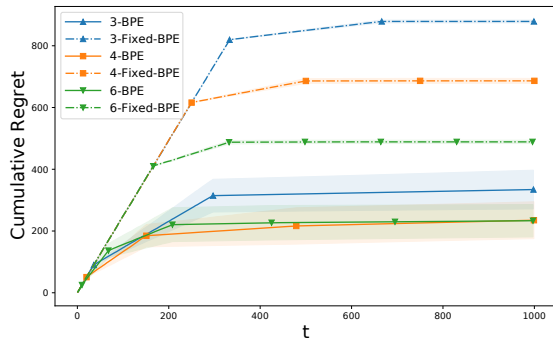Figure 3: f_1 with $\beta = 2$

Figure 4: f_2 with $\beta = 6$

Figure 5: BPE and Fixed-BPE on f_1 with $\beta = 2$
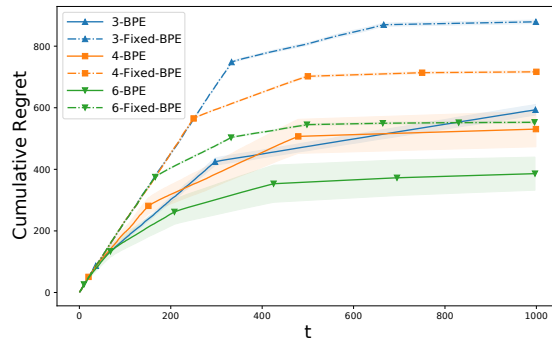

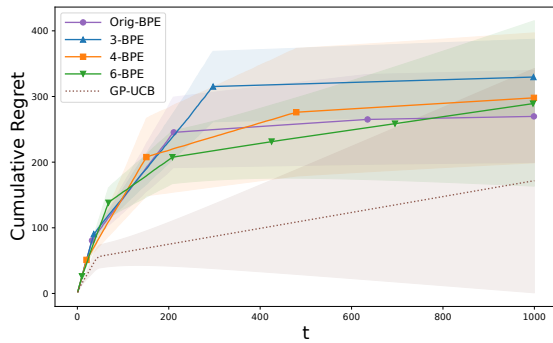Figure 6: BPE and Fixed-BPE on f_2 with $\beta = 6$

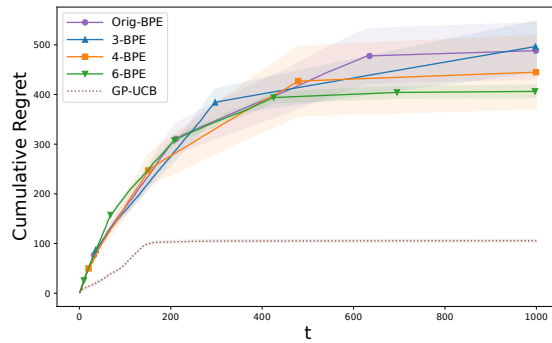Figure 7: f_1 with $\beta = 2$ and full posterior

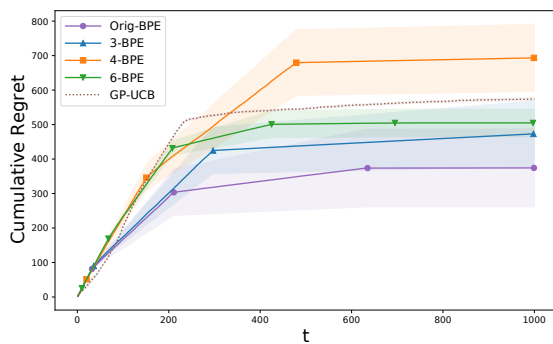Figure 8: f_2 with $\beta = 6$ and full posterior

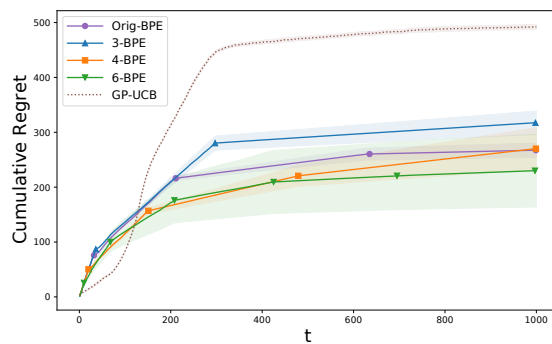Figure 9: f_1 with $\beta_i = 3\log(2i)$ and full posterior

Figure 10: f_2 with $\beta_i = 3\log(2i)$ and full posterior

various versions of BPE moving closer together, lower values of B having reduced regret, but the higher values of B having increased regret.

Finally, from Figures 9 and 10, when we use $\beta_i = 3\log(2i)$, the ordering of the algorithms can change significantly. This is because of two distinct factors impacting the algorithms in different ways: Increasing B has its obvious benefit of gathering information more frequently, but it also has the effect of making β_i grow larger than the ideal value. An extreme case of this is GP-UCB in Figure 10, where the regret is significantly higher due to β_i eventually growing as large as $3\log(2000) \approx 22.8$. Of course, further tuning β separately for each algorithm would alleviate this issue, but may not always be practical.

6 CONCLUSION

We introduced BPE algorithm for GP bandit optimization with few batches. When the number of batch is not specified by the problem, BPE achieves $O^*(\sqrt{T\gamma_T})$ cumulative regret with $O(\log\log T)$ batches. When the number of batches is a small constant specified in the problem, we proposed BPE with modified batch size arrangement. We also provided a lower bound for general batch algorithms with a given number of batches.

Acknowledgements

References

Cai, X., Gomes, S., and Scarlett, J. (2021). Lenient regret and good-action identification in Gaussian process bandits. In *Int. Conf. Mach. Learn. (ICML)*.

Cai, X. and Scarlett, J. (2021). On lower bounds for standard and robust Gaussian process bandit optimization. In *Int. Conf. Mach. Learn. (ICML)*.

Calandriello, D., Carratino, L., Lazaric, A., Valko, M., and Rosasco, L. (2019). Gaussian process optimization with adaptive sketching: Scalable and no regret. In *Conf. Learn. Theory (COLT)*.

Camilleri, R., Katz-Samuels, J., and Jamieson, K. (2021). High-dimensional experimental design and kernel bandits. In *Int. Conf. Mach. Learn. (ICML)*.

Cesa-Bianchi, N., Dekel, O., and Shamir, O. (2013). Online learning with switching costs and other adaptive adversaries. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*, volume 26.

Chowdhury, S. R. and Gopalan, A. (2017). On kernelized multi-armed bandits. In *Int. Conf. Mach. Learn. (ICML)*.

Contal, E., Buffoni, D., Robicquet, A., and Vayatis, N. (2013). *Machine Learning and Knowledge Discovery in Databases*, chapter Parallel Gaussian Process Optimization with Upper Confidence Bound and Pure Exploration, pages 225–240. Springer Berlin Heidelberg.

Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory*. John Wiley & Sons, Inc.

Desautels, T., Krause, A., and Burdick, J. W. (2014). Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization. *J. Mach. Learn. Res.*, 15(1):3873–3923.

Esfandiari, H., Karbasi, A., Mehrabian, A., and Mirrokni, V. (2021). Regret bounds for batched bandits. In *AAAI Conf. Art. Intel.*, volume 35, pages 7340–7348.

Janz, D., Burt, D. R., and González, J. (2020). Bandit optimisation of functions in the Matérn kernel RKHS. In *Int. Conf. Art. Intel. Stats. (AISTATS)*.

Kandasamy, K., Krishnamurthy, A., Schneider, J., and Póczos, B. (2018). Parallelised Bayesian optimisation via Thompson sampling. In *Int. Conf. Art. Intel. Stats. (AISTATS)*, pages 133–142.

Lizotte, D., Wang, T., Bowling, M., and Schuurmans, D. (2007). Automatic gait optimization with Gaussian process regression. In *Int. Joint. Conf. Art. Intel. (IJCAI)*, pages 944–949.

Mockus, J., Tiesis, V., and Zilinskas, A. (1978). The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2(117-129):2.

Nguyen, V., Rana, S., Gupta, S. K., Li, C., and Venkatesh, S. (2016). Budgeted batch Bayesian optimization. In *IEEE Int. Conf. Data Mining (ICDM)*, pages 1107–1112.

Rolland, P., Scarlett, J., Bogunovic, I., and Cevher, V. (2018). High-dimensional Bayesian optimization via additive models with overlapping groups. In *Int. Conf. Art. Intel. Stats. (AISTATS)*.

Salgia, S., Vakili, S., and Zhao, Q. (2021). A domain-shrinking based Bayesian optimization algorithm with order-optimal regret performance. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*.

Scarlett, J., Bogunovic, I., and Cevher, V. (2017). Lower bounds on regret for noisy Gaussian process bandit optimization. In *Conf. Learn. Theory (COLT)*.

Shekhar, S. and Javidi, T. (2020). Multi-scale zero-order optimization of smooth functions in an RKHS. https://arxiv.org/abs/2005.04832.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. (2010). Gaussian process optimization in the bandit setting: No regret and experimental design. In *Int. Conf. Mach. Learn. (ICML)*.

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294.

Vakili, S., Bouziani, N., Jalali, S., Bernacchia, A., and Shiu, D.-s. (2021a). Optimal order simple regret for Gaussian process bandits. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*.

Vakili, S., Khezeli, K., and Picheny, V. (2021b). On information gain and regret bounds in Gaussian process bandits. In *Int. Conf. Art. Intel. Stats. (AISTATS)*.

Vakili, S., Scarlett, J., Shiu, D.-s., and Bernacchia, A. (2022). Improved convergence rates for sparse approximation methods in kernel-based learning. https://arxiv.org/abs/2202.04005.

Valko, M., Korda, N., Munos, R., Flaounas, I., and Cristianini, N. (2013). Finite-time analysis of kernelised contextual bandits. In *Conf. Uncertainty in Art. Intel. (UAI)*.

Vanchinathan, H. P., Nikolic, I., De Bona, F., and Krause, A. (2014). Explore-exploit in top-N recommender systems via Gaussian processes. In *ACM Conf. Rec. Sys.*, pages 225–232.

Supplementary Material:
Gaussian Process Bandit Optimization with Few Batches

All citations in this appendix are to the reference list in the main body.

A   PROOFS OF MAIN RESULTS

A.1   Proof of Theorem 1 (Upper Bound with $O(\log \log T)$ Batches)

As discussed in the main text, we can apply Corollary 1 to any single batch due to the manner in which our algorithm ignores previous batches. To ensure valid confidence bounds simultaneously for all batches with probability at least $1 - \delta$, we further replace δ by $\frac{\delta}{B}$ therein, and apply the union bound. This yields $\beta = \left(\Psi + \frac{R}{\sqrt{\lambda}} \sqrt{2 \log \frac{|\mathcal{X}|B}{\delta}} \right)^2$, as stated in the description of the algorithm. We henceforth condition on the resulting confidence bounds remaining valid in all batches.

With such conditioning, we can deduce the theoretical upper bound on the overall cumulative regret by aggregating the cumulative regret of individual batches. For the first batch with $i = 1$ and $t \leq N_1$, the simple regret of any selected point is upper bounded as follows:

$$r_t = f(x^*) - f(x_t) \overset{(a)}{\leq} \mathrm{UCB}_1(x^*) - \mathrm{LCB}_1(x_t) \leq (\mu_0 + \sqrt{\beta}\sigma_0) - (\mu_0 - \sqrt{\beta}\sigma_0) \overset{(b)}{\leq} 2\sqrt{\beta},$$

where (a) uses the validity of the confidence bounds, and (b) follows since $\mu_0(\cdot) = 0$ and $\sigma_0(\cdot) = k(\cdot, \cdot) \leq 1$. Hence, with $N_1 = \lceil \sqrt{T} \rceil < \sqrt{T} + 1$, the cumulative regret of the first batch is

$$R^1 = \sum_{t=1}^{N_1} r_t \leq 2N_1\sqrt{\beta} = O(\sqrt{T\beta}). \tag{7}$$

From the second batch onwards with $i \geq 2$ and $t > N_1$, we make use of the following lemma, which is stated in a generic form (i.e., it should initially be read without regard to using batches).

Lemma 4. (Cai et al., 2021, Appendix B.5) *Suppose that we run an algorithm that, at each time t, samples the point x_t with the highest value of $\sigma_{t-1}(x_t)$. Moreover, suppose that after τ rounds of sampling, all points in the domain satisfy a confidence bound of the form $|f(x) - \mu_\tau(x)| \leq \sqrt{\beta}\sigma_\tau(x)$ for some $\beta > 0$. Then, for any point x whose UCB score is higher than the maximum LCB score, it must hold that*

$$\left| f(x^*) - f(x) \right| \leq 2\sqrt{\frac{C_1 \gamma_\tau \beta}{\tau}}, \tag{8}$$

where $C_1 = 8/\log(1 + \lambda^{-1})$, and x^* is any maximizer of f.

Since Lemma 4 is a known result, we do not provide a proof, but we briefly note that it follows easily from the following standard facts:

(i) As long as the confidence bounds are valid, x^* is not eliminated (in the sense of the UCB vs. LCB comparison mentioned above).

(ii) Under the maximum-variance selection rule, upper bounding the sampled point's variance amounts to upper bounding the variance of all points.

(iii) The sum of variances across all sampled points can be bounded in terms of γ_T using techniques dating back to (Srinivas et al., 2010) that are now very standard.

(iv) If the confidence width uniformly below η, then all non-eliminated points must have a function value within $O(\eta)$ of the maximum.

Using Lemma 4, we deduce that after the N_{i-1} points in the $(i-1)$-th batch are sampled, every point x_t in the i-th batch has regret r_t upper bounded as follows:

$$r_t \le 2\sqrt{\frac{C_1 \gamma_{N_{i-1}} \beta}{N_{i-1}}} \le 2\sqrt{\frac{C_1 \gamma_T \beta}{N_{i-1}}}, \tag{9}$$

where we used the fact that N_{i-1} points are sampled in the previous batch, and applied $\gamma_{N_{i-1}} \le \gamma_T$ by the monotonicity of γ_t.

The choice $N_i = \left\lceil \sqrt{T N_{i-1}} \right\rceil$ implies $N_i < \sqrt{T N_{i-1}} + 1$, and therefore

$$\frac{N_i}{\sqrt{N_{i-1}}} < \sqrt{T} + \frac{1}{\sqrt{N_{i-1}}} \le \sqrt{T} + \frac{1}{\sqrt{N_1}} \le \sqrt{T} + T^{-\frac{1}{4}},$$

which implies that the cumulative regret of batch i is

$$R^i = \sum_{t=t_{i-1}+1}^{t_i} r_t \le 2N_i \sqrt{\frac{C_1 \gamma_T \beta}{N_{i-1}}} \le 2\left(\sqrt{T} + T^{-\frac{1}{4}}\right)\sqrt{C_1 \gamma_T \beta} = O(\sqrt{T \gamma_T \beta}).$$

Combining the regret bounds for the batches, it holds with probability at least $1 - \delta$ that the cumulative regret incurred up to time T satisfies

$$R_T = R^1 + \sum_{i=2}^{B} R^i = O(B\sqrt{T \gamma_T \beta}) = O\big((\log \log T)\sqrt{T \gamma_T \beta}\big) = O^*(\Lambda \sqrt{T \gamma_T}),$$

where $\Lambda = \Psi + \sqrt{\log \frac{|\mathcal{X}|}{\delta}}$ (the $B = O(\log \log T)$ inside the logarithm can be absorbed into the $O^*(\cdot)$ notation, which hides dimension-independent log factors). The kernel-specific bounds then follow from (3) and (4).

A.2 Proof of Theorem 2 (Upper Bound with a Constant Number of Batches)

We again condition on the confidence bounds holding, and compute the regret upper bound for each individual batch. For the first batch $i = 1$, same as (7), we have

$$R^1 \le 2\sqrt{\beta}N_1 = O(\sqrt{\beta}N_1).$$

For batch $i = 2, \ldots, B - 1$, we have from Lemma 4 (similarly to (9)) that

$$R^i \le 2N_i \sqrt{\frac{C_1 \beta \gamma_{N_{i-1}}}{N_{i-1}}} \tag{10}$$

$$= O\left(N_i \sqrt{\frac{\beta \gamma_{N_{i-1}}}{N_{i-1}}}\right). \tag{11}$$

For the SE kernel, we proceed as follows:

$$R^i = O\left(N_i \sqrt{\frac{\beta \gamma_T}{N_{i-1}}}\right) \tag{12}$$

$$\le O^*\left(N_i \sqrt{\frac{\beta (\log T)^d}{N_{i-1}}}\right) \tag{13}$$

$$\le O^*(\sqrt{\beta}N_1), \tag{14}$$

where:

- (13) follows from (3);

- (14) follows since

$$N_i(\log T)^{\frac{d}{2}}(N_{i-1})^{-\frac{1}{2}}$$

$$= \left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta^i}{1-\eta^B}} (\log T)^d \right\rceil (\log T)^{d\eta}(N_{i-1})^{-\eta} \tag{15}$$

$$= \left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta}{1-\eta^B}} (\log T)^d \cdot \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta^{i-1}}{1-\eta^B}\eta} \right\rceil (\log T)^{d\eta}(N_{i-1})^{-\eta} \tag{16}$$

$$\leq \left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta}{1-\eta^B}} (\log T)^d \right\rceil \cdot \left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta^{i-1}}{1-\eta^B}\eta} \right\rceil (\log T)^{d\eta}(N_{i-1})^{-\eta} \tag{17}$$

$$\leq \left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta}{1-\eta^B}} (\log T)^d \right\rceil \cdot \left(\left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta^{i-1}}{1-\eta^B}\eta} (\log T)^{d\eta} \right\rceil + (\log T)^{d\eta} \right)(N_{i-1})^{-\eta} \tag{18}$$

$$\leq \left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta}{1-\eta^B}} (\log T)^d \right\rceil \cdot \left(\left\lceil \left(\frac{T}{(\log T)^d}\right)^{\frac{1-\eta^{i-1}}{1-\eta^B}} (\log T)^d \right\rceil^{\eta} + 1 + (\log T)^{d\eta} \right)(N_{i-1})^{-\eta} \tag{19}$$

$$= N_1\big((N_{i-1})^{\eta} + 1 + (\log T)^{d\eta}\big)(N_{i-1})^{-\eta} \tag{20}$$

$$= O(N_1), \tag{21}$$

and where:

- (15) follows from the definitions of N_i and $\eta = \frac{1}{2}$;
- (16) follows since $1 - \eta^i = (1-\eta) + (1-\eta^{i-1})\eta$;
- (17) follows since $\lceil ab \rceil \leq \lceil a \rceil \cdot \lceil b \rceil$ for $a, b > 0$;
- (18) follows since $\lceil a \rceil \cdot b \leq \lceil ab \rceil + b$ for $a, b > 0$;
- (19) follows since $\lceil a^{\eta} \rceil \leq \lceil a \rceil^{\eta} + 1$ for $a > 0, 0 < \eta < 1$;
- (20) follows from the definitions of N_1 and N_{i-1};
- (21) follows since $(\log T)^{d\eta} = \text{poly}(\log T)$ is asymptotically negligible compared to $(N_{i-1})^{\eta} = T^{\Theta(1)}$ (recall that d is considered constant with respect to T).

For the Matérn kernel, we proceed similarly:

$$R^i = O\left(N_i\sqrt{\frac{\beta\gamma_{N_{i-1}}}{N_{i-1}}}\right) \tag{22}$$

$$\leq O^*\left(N_i\sqrt{\frac{\beta(N_{i-1})^{1-2\eta}}{N_{i-1}}}\right) \tag{23}$$

$$= O^*\left(\sqrt{\beta}N_i(N_{i-1})^{-\eta}\right) \tag{24}$$

$$\leq O^*(\sqrt{\beta}N_1), \tag{25}$$

where:

- (23) follows from $O(\gamma_{N_{i-1}}) = O^*((N_{i-1})^{1-2\eta})$ by applying $\frac{d}{2\nu+d} = 1 - 2\eta$ with $\eta = \frac{\nu}{2\nu+d}$ in (4);

- (25) follows via similar steps to (15)–(21):

$$N_i(N_{i-1})^{-\eta} = \left\lceil T^{\frac{1-\eta^i}{1-\eta^B}} \right\rceil (N_{i-1})^{-\eta} \tag{26}$$

$$= \left\lceil T^{\frac{1-\eta}{1-\eta^B}} \cdot T^{\frac{1-\eta^{i-1}}{1-\eta^B}\eta} \right\rceil (N_{i-1})^{-\eta} \tag{27}$$

$$\leq \left\lceil T^{\frac{1-\eta}{1-\eta^B}} \right\rceil \cdot \left\lceil T^{\frac{1-\eta^{i-1}}{1-\eta^B}\eta} \right\rceil (N_{i-1})^{-\eta} \tag{28}$$

$$\leq \left\lceil T^{\frac{1-\eta}{1-\eta^B}} \right\rceil \left(\left\lceil T^{\frac{1-\eta^{i-1}}{1-\eta^B}} \right\rceil^{\eta} + 1 \right) (N_{i-1})^{-\eta} \tag{29}$$

$$= N_1 ((N_{i-1})^{\eta} + 1)(N_{i-1})^{-\eta} \tag{30}$$

$$\leq N_1 + (N_1)^{1-\eta} \tag{31}$$

$$= O(N_1). \tag{32}$$

For the last batch $i = B$ with the SE kernel, we have from (13) that

$$R^B = O^* \left(N_B \sqrt{\frac{\beta (\log T)^d}{N_{B-1}}} \right) \leq O^* \left(T \sqrt{\frac{\beta (\log T)^d}{N_{B-1}}} \right) = O^*(\sqrt{\beta} N_1),$$

where the last step follows by substituting $i = B$ into the right hand side of (15) and then substituting (21).

Similarly, for the last batch $i = B$ with the Matérn kernel, we have from (25) that

$$R^B = O^* \left(\sqrt{\beta} N_B (N_{B-1})^{-\eta} \right) \leq O^* \left(\sqrt{\beta} T (N_{B-1})^{-\eta} \right) = O^*(\sqrt{\beta} N_1),$$

where the last step follows by substituting $i = B$ into the right hand side of (26) and then applying (32).

We have thus shown that the assignment of batch sizes in Theorem 2 makes each batch have the same regret upper bound $O^*(\sqrt{\beta} N_1)$ for both kernels, and hence, we have the overall cumulative regret is

$$R_T = \sum_{i=1}^{B} R^i \leq O^*(B\sqrt{\beta} N_1) = O^*(\sqrt{\beta} N_1).$$

The proof is completed by substituting the choice of N_1.

A.3   Proof of Theorem 3 (Lower Bound with a Constant Number of Batches)

For the SE kernel with $\eta = \frac{1}{2}$, it is convenient to rewrite the lower bound in Corollary 2 as $R^m = \Omega\left(\left(\frac{T}{(\log T)^{d/2}} \right)^{b-\eta a} (\log T)^{d/2} \right)$. For any batch algorithm with time horizon T, number of batches B, and any pre-specified batch size arrangement, we will always fall into one of the B cases stated as follows:

1. With $t_B = T$ given, if $t_{B-1} < \left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta^{B-1}}{1-\eta^B}} (\log T)^{d/2}$, then by applying Corollary 2 with $a = \frac{1-\eta^{B-1}}{1-\eta^B}$ and $b = 1 = \frac{1-\eta^B}{1-\eta^B}$, there exists $f \in \mathcal{F}_k(\Psi)$ such that $R_T \geq R^B = \Omega\left(\left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta}{1-\eta^B}} (\log T)^{d/2} \right)$ with probability at least $\frac{3}{4}$.

2. Otherwise, if $t_{B-1} \geq \left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta^{B-1}}{1-\eta^B}} (\log T)^{d/2}$ and $t_{B-2} < \left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta^{B-2}}{1-\eta^B}} (\log T)^{d/2}$, then by applying Corollary 2 with $a = \frac{1-\eta^{B-2}}{1-\eta^B}$ and $b = \frac{1-\eta^{B-1}}{1-\eta^B}$, there exists $f \in \mathcal{F}_k(\Psi)$ such that $R_T \geq R^{B-1} = \Omega\left(\left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta}{1-\eta^B}} (\log T)^{d/2} \right)$ with probability at least $\frac{3}{4}$, since $b - \frac{a}{2} = b - \eta a = \frac{1-\eta}{1-\eta^B}$.

3. Similar to steps 1 and 2, for $m = B - 2, B - 3, \ldots, 2$, we iteratively argue as follows: If $t_m \geq \left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta^m}{1-\eta^B}} (\log T)^{d/2}$ and $t_{m-1} < \left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta^{m-1}}{1-\eta^B}} (\log T)^{d/2}$, then by applying Corollary 2 with $a = \frac{1-\eta^{m-1}}{1-\eta^B}$ and $b = \frac{1-\eta^m}{1-\eta^B}$, there exists $f \in \mathcal{F}_k(\Psi)$ such that $R_T \geq R^m = \Omega\left(\left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta}{1-\eta^B}} (\log T)^{d/2} \right)$ with probability at least $\frac{3}{4}$.

4. Now the only case remaining is when $t_1 = N_1 \geq \left(\frac{T}{(\log T)^{d/2}} \right)^{\frac{1-\eta}{1-\eta^B}} (\log T)^{d/2}$. The idea here is that the regret of points in the first batch is typically $\Omega(1)$ due to the fact that no observations have been made. In more

detail, we can consider f being any function with a clear peak of height $\Theta(1)$, such that the majority of the domain's points incur regret $\Omega(1)$. By considering a class of functions of this type[7] with the peak location shifted throughout the domain, we readily obtain that there exists $f \in \mathcal{F}_k(\Psi)$ such that $R^1 = \Omega(N_1)$, and hence $R_T \geq R^1 = \Omega\left(\left(\frac{T}{(\log T)^{d/2}}\right)^{\frac{1-\eta}{1-\eta^B}}(\log T)^{d/2}\right)$, with probability at least $\frac{3}{4}$.

Since $\frac{d}{2}\left(1 - \frac{1-\eta}{1-\eta^B}\right) = \frac{d}{2}\left(\frac{\eta - \eta^B}{1-\eta^B}\right)$, we have $R_T = \Omega\left(\left(\frac{T}{(\log T)^{d/2}}\right)^{\frac{1-\eta}{1-\eta^B}}(\log T)^{d/2}\right) = \Omega\left(T^{\frac{1-\eta}{1-\eta^B}}(\log T)^{\frac{d(\eta-\eta^B)}{2(1-\eta^B)}}\right)$ for some $f \in \mathcal{F}_k(\Psi)$ with probability at least $\frac{3}{4}$ in all cases, and the claimed lower bound follows.

Similarly, for the Matérn kernel with $\eta = \frac{\nu}{2\nu+d}$, the lower bound in Corollary 2 can be rewritten as $R^m = \Omega(T^{b-\eta a})$. For any batch algorithm with time horizon T, number of batches B, and any pre-specified batch size arrangement, we will always fall into one of the B cases stated as follows:

1. With $t_B = T$ given, if $t_{B-1} < T^{\frac{1-\eta^{B-1}}{1-\eta^B}}$, then by applying Corollary 2 with $a = \frac{1-\eta^{B-1}}{1-\eta^B}$ and $b = 1 = \frac{1-\eta^B}{1-\eta^B}$, there exists $f \in \mathcal{F}_k(\Psi)$ such that $R_T \geq R^B = \Omega\left(T^{\frac{1-\eta}{1-\eta^B}}\right)$ with probability at least $\frac{3}{4}$.

2. Otherwise, if $t_{B-1} \geq T^{\frac{1-\eta^{B-1}}{1-\eta^B}}$ and $t_{B-2} < T^{\frac{1-\eta^{B-2}}{1-\eta^B}}$, then by applying Corollary 2 with $a = \frac{1-\eta^{B-2}}{1-\eta^B}$ and $b = \frac{1-\eta^{B-1}}{1-\eta^B}$, there exists $f \in \mathcal{F}_k(\Psi)$ such that $R_T \geq R^{B-1} = \Omega\left(T^{\frac{1-\eta}{1-\eta^B}}\right)$ with probability at least $\frac{3}{4}$.

3. Similar to steps 1 and 2, for $m = B-2, B-3, \ldots, 2$, we iteratively argue as follows: If $t_m \geq T^{\frac{1-\eta^m}{1-\eta^B}}$ and $t_{m-1} < T^{\frac{1-\eta^{m-1}}{1-\eta^B}}$, then by applying Corollary 2 with $a = \frac{1-\eta^{m-1}}{1-\eta^B}$ and $b = \frac{1-\eta^m}{1-\eta^B}$, there exists $f \in \mathcal{F}_k(\Psi)$ such that $R_T \geq R^m = \Omega\left(T^{\frac{1-\eta}{1-\eta^B}}\right)$ with probability at least $\frac{3}{4}$.

4. Now the only case remaining is when $t_1 = N_1 \geq T^{\frac{1-\eta}{1-\eta^B}}$. In this case, by the same argument as the SE kernel above, there exists $f \in \mathcal{F}_k(\Psi)$ such that $R_T \geq R^1 = \Omega\left(T^{\frac{1-\eta}{1-\eta^B}}\right)$ for some $f \in \mathcal{F}_k(\Psi)$ with probability at least $\frac{3}{4}$.

It follows that $R_T = \Omega\left(T^{\frac{1-\eta}{1-\eta^B}}\right)$ for some $f \in \mathcal{F}_k(\Psi)$ with probability at least $\frac{3}{4}$ in all cases, and the claimed lower bound follows.

B DISCUSSION ON LOWER BOUNDS WITH ADAPTIVELY-CHOSEN BATCH LENGTHS

To discuss the difficulties in establishing tight lower bounds when the batches sizes can be chosen adaptively (as opposed to being pre-specified), it is useful to recap how the algorithm-independent lower bounds are proved in (Scarlett et al., 2017; Cai and Scarlett, 2021).

B.1 Background

The high-level idea in the above-mentioned works is to form a lower bound based on the difficulty of locating a small narrow bump in $[0,1]^d$, where the peak of the bump is 2ϵ for some parameter $\epsilon > 0$, but the vast majority of the space has $f(x) \approx 0$ (or even $f(x) = 0$). The RKHS norm constraint limits how narrow the bump can be, and accordingly how many of these functions can be "packed" into the space while keeping the bump parts non-overlapping.

For the SE and Matérn kernels, the quantities that arise are as follows:

---

[7]For brevity, we have not described a specific "hard" function class here, but formally, the class of functions used in previous lower bounds (Scarlett et al., 2017; Cai and Scarlett, 2021) would suffice. The idea is that in the same way that Corollary 2 derives a cumulative regret bound from Lemma 3's simple regret bound, here we get an $\Omega(N_1)$ cumulative regret bound directly from the trivial $\Omega(1)$ simple regret bound when *no points are observed*.

- For the SE kernel, the bump width is on the order of $w \sim \frac{1}{\sqrt{\log \frac{1}{\epsilon}}}$, and so the total number of bumps is on the order of $M \sim \left(\log \frac{1}{\epsilon} \right)^{d/2}$ (constants are omitted in this informal discussion).

- For the Matérn kernel, the bump width is on the order of $w \sim \epsilon^{1/\nu}$, yielding $M \sim \left(\frac{1}{\epsilon} \right)^{d/\nu}$.

The idea is then that since there are M possible bump locations but the function only peaks at 2ϵ and is nearly zero throughout the domain, around $T \sim \frac{M}{\epsilon^2}$ samples are needed to locate the bump (and thus attain simple regret at most ϵ). Solving this formula for ϵ gives the lower bound on the simple regret as a function of T.

B.2 Difficulty in the Batch Setting

The proof of Theorem 3 is based on a recursive argument based on how long the batch lengths are:

- If the final batch starts too early, then the information gathered in the first $B - 1$ batches is insufficient to locate a height-(ϵ_{B-1}) bump (for some suitably-chosen ϵ_{B-1}), so every action played in the final batch is likely to incur $\Omega(\epsilon_{B-1})$ regret.

- If the final batch does start sufficiently late to avoid the previous case, but the second-last batch starts too early, then the information gathered in the first $B - 2$ batches is insufficient to locate a height-(ϵ_{B-2}) bump, amounting to $\Omega(\epsilon_{B-2})$ regret for actions in the second-last batch.

- The previous dot point continues recursively, and the final case is that if the first batch starts too late, then due to having no prior information for the first batch, each selected action will incur $\epsilon_0 = \Omega(1)$ regret.

The difficulty is that *the lower bound on regret in each of these cases corresponds to a different "hard subset" of functions*. Specifically, the difficult case for the later batches is a function class with a larger number of functions consisting of shorter and narrower bumps, whereas the difficult case for the earlier batches is a function class with a smaller number of functions consisting of relatively higher and wider bumps.

If the batch lengths are pre-specified, then we can readily choose the hard subset for the appropriate case depending on those lengths. However, if the batch sizes are chosen adaptively, then the argument appears to be more difficult, because in principle the algorithm could suitably adapt the batch lengths to overcome the difficulty of optimizing a single *a priori* specified hard class.

B.3 Plausibility Argument

Here we outline a plausibility argument for Theorem 3 remaining true even in the case of adaptively-chosen batch sizes. To do so, we let $t_1^*, \ldots, t_B^*$ be the "critical" ending times of the batches (with $t_B^* = T$) dictated by the proof of Theorem 3, e.g., $t_1^* \sim T^{\frac{1-\eta}{1-\eta^B}}$ for the Matérn kernel according to the item 4 at the end of Appendix A.3. In addition, we let $\mathcal{F}_1, \ldots, \mathcal{F}_B$ be the hard function classes associated with batches $1, \ldots, B$, as discussed above.

The plausibility argument is as follows. Suppose that the underlying function comes from $\mathcal{F}_1 \cup \ldots \mathcal{F}_B$, but is otherwise unknown. Then:

- To handle the possibility that the function is from $\mathcal{F}_1$, the first batch length must be no higher than t_1^*.

- After observing the first batch of size at most t_1^*, if the function was not from $\mathcal{F}_1$, then no matter what class among $\mathcal{F}_2, \ldots, \mathcal{F}_B$ the function was from, not enough samples have been taken to know where the function's bump lies. This suggests that the algorithm still has no way of reliably distinguishing between $\mathcal{F}_2, \ldots, \mathcal{F}_B$, and it must prepare for the possibility that the function lies in $\mathcal{F}_2$. Thus, the second batch should finish no later than time t_2^*.

- This argument continues recursively until the $(B - 1)$-th batch finishes no later than time t_{B-1}^*, but even in this case, the final batch incurs enough regret to produce the desired lower bound.

Formalizing this argument is left for future work.