Journal of Machine Learning Research 15 (2014) 4053-4103 Submitted 4/13; Revised 6/14; Published 12/14

Parallelizing Exploration-Exploitation Tradeoffs in Gaussian Process Bandit Optimization

Thomas Desautels* TDESAUTELS@GATSBY.UCL.AC.UK
Gatsby Computational Neuroscience Unit
University College London
Alexandra House, 17 Queen Square, London WC1N 3AR, UK

Andreas Krause KRAUSEA@ETHZ.CH
Department of Computer Science
ETH Zurich
Universitätstrasse 6, 8092 Zürich, Switzerland

Joel W. Burdick JWB@ROBOTICS.CALTECH.EDU
Department of Mechanical Engineering
California Institute of Technology
1200 E California Blvd., Pasadena, CA 91125, USA

Editor: Alexander Rakhlin

Abstract

How can we take advantage of opportunities for experimental parallelization in exploration-exploitation tradeoffs? In many experimental scenarios, it is often desirable to execute experiments simultaneously or in batches, rather than only performing one at a time. Additionally, observations may be both noisy and expensive. We introduce Gaussian Process Batch Upper Confidence Bound (GP-BUCB), an upper confidence bound-based algorithm, which models the reward function as a sample from a Gaussian process and which can select batches of experiments to run in parallel. We prove a general regret bound for GP-BUCB, as well as the surprising result that for some common kernels, the asymptotic average regret can be made independent of the batch size. The GP-BUCB algorithm is also applicable in the related case of a delay between initiation of an experiment and observation of its results, for which the same regret bounds hold. We also introduce Gaussian Process Adaptive Upper Confidence Bound (GP-AUCB), a variant of GP-BUCB which can exploit parallelism in an adaptive manner. We evaluate GP-BUCB and GP-AUCB on several simulated and real data sets. These experiments show that GP-BUCB and GP-AUCB are competitive with state-of-the-art heuristics.[1]

Keywords: Gaussian process, upper confidence bound, batch, active learning, regret bound

1. Introduction

Many problems require optimizing an unknown reward function, from which we can only obtain noisy observations. A central challenge is choosing actions that both explore (es-

∗. This research was carried out while TD was a student at the California Institute of Technology.

1. A previous version of this work appeared in the Proceedings of the 29th International Conference on Machine Learning, 2012.

timate) the function and exploit our knowledge about likely high reward regions in the function's domain. Carefully calibrating this exploration–exploitation tradeoff is especially important in cases where the experiments are costly, e.g., when each experiment takes a long time to perform and the time budget available for experiments is limited. In such settings, it may be desirable to execute several experiments in parallel. By parallelizing the experiments, substantially more information can be gathered in the same time-frame; however, future actions must be chosen without the benefit of intermediate results. One might conceptualize these problems as choosing "batches" of experiments to run simultaneously. The challenge is to assemble batches of experiments that both explore the function and exploit by focusing on regions with high estimated value.

Two key, interrelated questions arise: the *computational* question of how one should efficiently choose, out of the combinatorially large set of possible batches, those that are most effective; and the *statistical* question of how the algorithm's performance depends on the size of the batches (i.e., the degree of informational parallelism). In this paper, we address these questions by presenting GP-BUCB and GP-AUCB; these are novel, efficient algorithms for selecting batches of experiments in the Bayesian optimization setting where the reward function is modeled as a sample from a Gaussian process prior or has low norm in the associated Reproducing Kernel Hilbert Space.

In more detail, we provide the following main contributions:

- We introduce GP-BUCB, a novel algorithm for selecting actions to maximize reward in large-scale exploration-exploitation problems. GP-BUCB accommodates parallel or batch execution of the actions and the consequent observation of their reward. GP-BUCB may also be used in the setting of a bounded delay between initiation of an action and observation of its reward.
- We also introduce GP-AUCB, an algorithm which adaptively exploits parallelism to choose batches of actions, the sizes of which are limited by the conditional mutual information gained therein; this limit is such that the batch sizes are small when the algorithm selects actions for which it knows relatively little about the reward. Conversely, batch sizes may be large when the reward function is well known for the actions selected. We show that this adaptive parallelism is effective and can easily be parameterized using pre-defined limits.
- We prove sublinear bounds on the cumulative regret incurred by algorithms of a general class, including GP-BUCB and GP-AUCB, that also imply bounds on their rates of convergence.
- For some common kernels, we show that if the problem is initialized by making observations corresponding to an easily selected and provably bounded set of queries, the regret of GP-BUCB can be bounded to a constant multiplicative factor of the known regret bounds of the fully sequential GP-UCB algorithm of Srinivas et al. (2010, 2012). This implies (near-)linear speedup in the asymptotic convergence rates through parallelism.
- We demonstrate how execution of many UCB algorithms, including the GP-UCB, GP-BUCB, and GP-AUCB algorithms, can be drastically accelerated by *lazily evaluating* the posterior variance. This technique does not result in any loss in accuracy.

- We evaluate GP-BUCB and GP-AUCB on several synthetic benchmark problems, as well as two real data sets, respectively related to automated vaccine design and therapeutic spinal cord stimulation. We show that GP-BUCB and GP-AUCB are competitive with state-of-the-art heuristics for parallel Bayesian optimization. Under certain circumstances, GP-BUCB and GP-AUCB are competitive with sequential action selection under GP-UCB, despite having to cope with the disadvantage of delayed feedback.
- We consider more complex notions of execution cost in the batch and delay settings and identify areas of this cost and performance space where our algorithms make favorable tradeoffs and are therefore especially suitable for practical applications.

In the remainder of the paper, we first review the literature (Section 2) and formally describe the problem setting (Section 3). In the next section, we describe the GP-BUCB algorithm, present the main regret bound, which applies to a general class of algorithms using an upper confidence bound decision rule, and present corollaries bounding the regret of GP-BUCB and initialized GP-BUCB (Section 4). We extend this analysis to GP-AUCB, providing a regret bound for that algorithm, and discuss different possible stopping conditions for similar algorithms (Section 5). Next, we introduce the notion of lazy variance calculations (Section 6). We compare our algorithms' performance with each other and with several other algorithms across a variety of problem instances, including two real data sets (Section 7). Finally, we present our conclusions (Section 8).

2. Related Work

Our work builds on ideas from bandits, Bayesian optimization, and batch selection. In the following, we briefly review the literature in each of these areas.

2.1 Multi-armed Bandits

Exploration-exploitation tradeoffs have been classically studied in context of multi-armed bandit problems. These are sequential decision tasks where a single action is taken at each round, and a corresponding (possibly noisy) reward is observed. Early work has focused on the case of a finite number of candidate actions (arms), a total budget of actions which is at least as large as the number of arms, and payoffs that are independent across the arms (Robbins, 1952). In this setting, under some strong assumptions, optimal policies can be computed (Gittins, 1979). Optimistic allocation of actions according to upper-confidence bounds (UCB) on the payoffs has proven to be particularly effective (Auer et al., 2002). In many applications, the set of candidate actions is very large (or even infinite). In such settings, dependence between the payoffs associated with different decisions must be modeled and exploited. Various methods of introducing dependence include bandits with linear (Dani et al., 2008; Abernethy et al., 2008; Abbasi-Yadkori et al., 2011) or Lipschitz-continuous payoffs (Kleinberg et al., 2008; Bubeck et al., 2008), or bandits on trees (Kocsis and Szepesvári, 2006). In this paper we pursue a Bayesian approach to bandits, where fine-grained assumptions on the regularity of the reward function can be imposed through proper choice of the prior distribution over the payoff function. Concretely, we focus on Gaussian process priors, as considered by Srinivas et al. (2010).

2.2 Bayesian Optimization

The exploration-exploitation tradeoff has also been studied in Bayesian global optimization and response surface modeling, where Gaussian process (GP, see Rasmussen and Williams, 2006) models are often used due to their flexibility in incorporating prior assumptions about the payoff function's structure (Brochu et al., 2010). In addition to a model of the payoff function, an algorithm must have a method for selecting the next observation. Several bandit-like heuristics, such as Maximum Expected Improvement (Jones et al., 1998), Maximum Probability of Improvement (Mockus, 1989), Knowledge Gradient (Ryzhov et al., 2012), and upper-confidence-based methods (Cox and John, 1997), have been developed to balance exploration with exploitation and have been successfully applied in learning problems (e.g., Lizotte et al., 2007). In contrast, the Entropy Search algorithm of Hennig and Schuler (2012) seeks to take the action that will greedily decrease future losses, a less bandit-like and more optimization-focused heuristic. Recently, Srinivas et al. (2010, 2012) analyzed GP-UCB, an algorithm for this setting based on upper-confidence bound sampling, and proved bounds on its cumulative regret, and thus convergence rates for Bayesian global optimization. We build on this foundation and generalize it to the parallel setting.

2.3 Parallel Selection

To enable parallel selection, one must account for the delay between decisions and observations. Most existing approaches that can deal with such delay result in a multiplicative increase in the cumulative regret as the delay grows. Only recently, Dudik et al. (2011) demonstrated that it is possible to obtain regret bounds that only increase *additively* with the delay (i.e., the penalty becomes negligible for large numbers of decisions). However, the approach of Dudik et al. only applies to contextual bandit problems with finite decision sets, and thus not to settings with complex (even nonparametric) payoff functions. Similarly, contemporary work by Joulani et al. (2013) develops a meta-algorithm for converting sequential bandit algorithms to the delayed, finite decision set environment. While this algorithm has regret bounds which only increase additively with batch size, it does not generalize to the case of infinitely large decision sets and, by construction, does not take advantage of knowledge of pending observations, leading to redundant exploration, of particular concern when individual observations are expensive.

In contrast to these theoretical developments for finite bandits, there has been heuristic work on parallel Bayesian global optimization using GPs, e.g., by Ginsbourger et al. (2010). The state of the art is the *simulation matching* algorithm of Azimi et al. (2010), which uses the posterior of the payoff function at the beginning of the batch to simulate observations that the sequential algorithm would encounter if it could receive feedback during the batch, obtaining a number of Monte Carlo samples over future behaviors of the sequential algorithm. These Monte Carlo samples are then aggregated into a batch of observations which is intended to "closely match" the set of actions that would be taken by the sequential algorithm if it had been run with sequential feedback. To our knowledge, no theoretical results regarding the regret or convergence of this algorithm exist. We experimentally compare with this approach in Section 7. Azimi et al. (2012a) recently extended this construction to the batch classification setting.

Azimi et al. (2012b) also propose a very different algorithm that adaptively chooses the level of parallelism it will allow. This is done in a manner which depends on the expected prediction error between the posterior constructed with the simulated observations in the batch in progress versus the true posterior that would be available assuming the observations had actually been obtained. We also compare against this adaptive algorithm in Section 7.

Recently, Chen and Krause (2013) investigated batch-mode active learning using the notion of adaptive submodular functions. In contrast to our work, their approach focuses on active learning for estimation, which does not involve exploration–exploitation tradeoffs.

3. Problem Setting and Background

We wish to take a sequence of *actions* (or equivalently, make decisions) $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T \in D$, where D is the *decision set*, which is often (but not necessarily) a compact subset of $\mathbb{R}^d$. The subscript denotes the *round* in which that action was taken; each round is an opportunity for the algorithm to take one action. For each action $\boldsymbol{x}_t$, we observe a noisy scalar reward $y_t = f(\boldsymbol{x}_t) + \varepsilon_t$, where $f : D \to \mathbb{R}$ is an unknown function modeling the expected payoff $f(\boldsymbol{x})$ for each action $\boldsymbol{x}$. For now we assume that the noise variables ε_t are i.i.d. Gaussian with known variance σ_n^2, i.e., $\varepsilon_t \sim \mathcal{N}(0, \sigma_n^2)$, $\forall t \geq 1$. This assumption will be relaxed later in one of the cases of our main theorem. If the actions $\boldsymbol{x}_t$ are selected one at a time, each with the benefit of all observations $y_1, \ldots, y_{t-1}$ corresponding to previous actions $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}$, we shall refer to this case as the *strictly sequential* setting. In contrast, the main problem tackled in this paper is the challenging setting where action $\boldsymbol{x}_t$ must be selected using only observations $y_1, \ldots, y_{t'}$, where often $t' < t - 1$. Thus, less information is available for choosing actions as compared to the strictly sequential setting.

In selecting these actions, we wish to maximize the cumulative reward $\sum_{t=1}^{T} f(\boldsymbol{x}_t)$. Defining the *regret* of action $\boldsymbol{x}_t$ as $r_t = [f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t)]$, where $\boldsymbol{x}^* \in \boldsymbol{X}^* = \operatorname{argmax}_{\boldsymbol{x} \in D} f(\boldsymbol{x})$ is an optimal action (assumed to exist, but not necessarily to be unique), we may equivalently think of maximizing the cumulative reward as minimizing the *cumulative regret*

$$R_T = \sum_{t=1}^{T} r_t.$$

By minimizing the regret, we ensure progress toward optimal actions uniformly over T. In fact, the *average regret*, R_T/T, is a natural upper bound on the suboptimality of the best action considered so far, i.e., $R_T/T \geq \min_{t \in 1,\ldots,T} [f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t)]$ (where this minimum is often called the *simple regret*, Bubeck et al., 2009). Therefore bounds on the average regret imply convergence rates for global optimization. It is particularly desirable to show that R_T is sublinear, i.e., $o(T)$, such that the average regret (and thus the minimum regret) goes to zero for large T; an algorithm with this property is described as being "no-regret."

In Section 3.1, we formally define the problem setting of parallel selection. Sections 3.2 and 3.3 introduce mathematical background necessary for our analysis. Section 3.4 describes the GP-UCB algorithm and discusses why some simple attempts at generalizing it to the parallel setting are insufficient, setting the stage for GP-BUCB, the subject of Section 4.

3.1 The Problem: Parallel or Delayed Selection

In many applications, at time τ, we wish to select a *batch* of actions, e.g., $\boldsymbol{x}_\tau, ..., \boldsymbol{x}_{\tau+B-1}$, where B is the size of the batch, to be evaluated in parallel. One natural application is the design of high-throughput experiments, where several experiments are performed in parallel, but feedback is only received after the experiments have concluded. In other settings, the feedback is delayed. We can model both situations by selecting actions sequentially; however when choosing $\boldsymbol{x}_t$ in round t, we can only make use of the feedback obtained in rounds $1, \ldots, t'$, for some $t' \leq t-1$. Formally, we assume there is some mapping $\text{fb} : \mathbb{N} \to \{\mathbb{N}, 0\}$ (where $\mathbb{N}$ denotes the positive integers) such that $\text{fb}[t] \leq t-1, \ \forall t \in \mathbb{N}$, and when selecting an action at time t, we can use feedback up to and including round $\text{fb}[t]$. If $\text{fb}[t] = 0$, no observation information is available.

Here and in most of the remainder of the paper, we concentrate primarily on this perspective on parallelism, which we term the *pessimistic view*, in which we consider the problem of *coping effectively under inferior feedback*. Intuitively, given feedback such that $\text{fb}[t] \leq t-1$ and often $\text{fb}[t] < t-1$, an algorithm should be expected to underperform relative to the strictly sequential algorithm, which obtains feedback according to $\text{fb}[t] = t-1$. This view provides a natural benchmark; success is performing nearly as well as the strictly sequential algorithm, despite the disadvantageous feedback. The contrasting *optimistic view*, in which parallelism may confer an advantage over strictly sequential algorithms via the ability to take more than one action simultaneously, is equivalent to the pessimistic view via a reparameterization of time, if batches are constructed sequentially; the difference between the two is fundamentally the philosophical primacy of decision-making in the pessimistic view and the experimental process in the optimistic view. We examine our results from the optimistic perspective in Section 7.3 and in Figure 7. Unfortunately, this optimistic view of parallelism presents difficulties when comparing algorithms; there is less clearly a benchmark for comparing the regret suffered by two algorithms which have submitted the same number of batches but use different levels of parallelism, since they may at any time have different numbers of observations. We thus concentrate our analytical and experimental approach on the pessimistic view, while remaining motivated by its optimistic counterpart.

Different specifications of the feedback mapping $\text{fb}[t]$ can model a variety of realistic scenarios. As noted above, setting $B = 1$ and $\text{fb}[t] = t-1$ corresponds to the non-delayed, strictly sequential setting in which a single action is selected and the algorithm waits until the corresponding observation is returned before selecting the succeeding action. The *simple batch* setting, in which we wish to select batches of size B, can be captured by setting $\text{fb}[t]_{SB} = \lfloor (t-1)/B \rfloor B$, i.e.,

$$\text{fb}[t]_{SB} = \begin{cases} 0 & : t \in \{1, \ldots, B\} \\ B & : t \in \{B+1, \ldots, 2B\} \\ 2B & : t \in \{2B+1, \ldots, 3B\} \\ \vdots & \end{cases}.$$

Note that in the batch case, the time indexing within the batch is a matter of algorithmic construction, since the batch is built in a sequential fashion, but actions are initiated simultaneously and observations are received simultaneously. If we wish to formalize the problem of selecting actions when feedback from those actions is delayed by exactly B rounds, the

simple delay setting, we can simply define this feedback mapping as $\text{fb}[t]_{SD} = \max\{t-B, 0\}$. Note that in both the simple batch and delay cases, $B = 1$ is the strictly sequential case. In comparing these two simple cases for equal values of B, we observe that $\text{fb}[t]_{SB} \geq \text{fb}[t]_{SD}$, that is, the set of observations available in the simple batch case for selecting the tth action is always at least as large as in the simple delay case, suggesting that the delay case is in some sense "harder" than the batch case. As we will see, however, the regret bounds presented in this paper may be expressed in terms of the maximal number of pending observations (i.e., those which have been initiated, but are still incomplete), which is $B-1$ in both of these settings, resulting in unified proofs and regret bounds for the two cases.

More complex cases may also be described using a feedback mapping. For example, we may be interested in executing B experiments in parallel, where we can start a new experiment as soon as one finishes, but the length of each experiment is variable; this translates to a more complex delay setting in which the algorithm has a queue of pending observations of some finite size B and checks at each round to see whether the queue is full. If the queue is not full, the algorithm submits an action, and if it is full, it "balks," i.e., does not submit an action and continues waiting for room to open within the queue. This is a natural description of an agent which periodically monitors slow experimental processes and takes action when it discovers they have finished. Since the algorithm only selects a new action when the queue is not full, there can be at most $B-1$ pending observations at the time a new action is selected, as in the simple batch and delay cases. Again, the maximum number of pending observations is the key to bounding the regret.

Since the level of difficulty of a variety of settings may be described in terms of the maximum number of pending observations when selecting any action (which we set to be $B-1$), in our development of GP-BUCB and initialized GP-BUCB in Sections 4.4 and 4.5, we only assume that the mapping $\text{fb}[t]$ is specified as part of the problem instance and $t - \text{fb}[t] \leq B$ for a known constant B. Importantly, our algorithms do not need to know the full feedback mapping ahead of time. It suffices if $\text{fb}[t]$ is revealed to the algorithms at each time t.

3.2 Modeling f via Gaussian Processes

Regardless of when feedback is obtained, if we are to turn a finite number of observations into useful inference about the payoff function f, we will have to make assumptions about its structure. In particular, for large (possibly infinite) decision sets D there is no hope to do well, i.e., incur little regret or even simply converge to an optimal action, if we do not make any assumptions. For good performance, we must choose a regression model which is both simple enough to be learned and expressive enough to capture the relevant behaviors of f. One effective formalism is to model f as a sample from a Gaussian process[2] (GP) prior. A GP is a probability distribution across a class of—typically smooth—functions, which is parameterized by a kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$, which characterizes the smoothness of f, and a mean function $\mu(\boldsymbol{x})$. In the remainder of this section, we assume $\mu(\boldsymbol{x}) = 0$ for notational convenience, without loss of generality. We often also assume that $k(\boldsymbol{x}, \boldsymbol{x}) \leq 1$, $\forall \boldsymbol{x} \in D$, i.e., that the kernel is normalized; results obtained using this assumption can be generalized to any case where $k(\boldsymbol{x}, \boldsymbol{x})$ has a known bound. We write $f \sim \mathcal{GP}(\mu, k)$ to denote that

2. See Rasmussen and Williams (2006) for a thorough treatment.

we model f as sampled from such a GP. If noise is i.i.d. Gaussian and the distribution of f is conditional on a vector of observations $\boldsymbol{y}_{1:t-1} = [y_1, ..., y_{t-1}]^T$ corresponding to actions $\boldsymbol{X}_{t-1} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_{t-1}]^T$, one obtains a Gaussian posterior distribution $f(\boldsymbol{x})|\boldsymbol{y}_{1:t-1} \sim \mathcal{N}(\mu_{t-1}(\boldsymbol{x}), \sigma^2_{t-1}(\boldsymbol{x}))$ for each $\boldsymbol{x} \in D$, where

$$\mu_{t-1}(\boldsymbol{x}) = K(\boldsymbol{x}, \boldsymbol{X}_{t-1})[K(\boldsymbol{X}_{t-1}, \boldsymbol{X}_{t-1}) + \sigma_n^2 I]^{-1}\boldsymbol{y}_{1:t-1} \text{ and} \tag{1}$$

$$\sigma^2_{t-1}(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x}) - K(\boldsymbol{x}, \boldsymbol{X}_{t-1})[K(\boldsymbol{X}_{t-1}, \boldsymbol{X}_{t-1}) + \sigma_n^2 I]^{-1}K(\boldsymbol{x}, \boldsymbol{X}_{t-1})^T. \tag{2}$$

In the above, $K(\boldsymbol{x}, \boldsymbol{X}_{t-1})$ denotes the row vector of kernel evaluations between $\boldsymbol{x}$ and the elements of $\boldsymbol{X}_{t-1}$, the set of actions taken in the past, and $K(\boldsymbol{X}_{t-1}, \boldsymbol{X}_{t-1})$ is the matrix of kernel evaluations where $[K(\boldsymbol{X}_{t-1}, \boldsymbol{X}_{t-1})]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\forall\, \boldsymbol{x}_i, \boldsymbol{x}_j \in \boldsymbol{X}_{t-1}$, i.e., the covariance matrix of the values of f over the set so far observed. Since Equations (1) and (2) can be computed efficiently, closed-form posterior inference is computationally tractable in a GP distribution via linear algebraic operations.

3.3 Conditional Mutual Information

A number of information theoretic quantities will be essential to the analysis of the algorithms presented in this paper. In particular, we are interested in the mutual information $I(f; \boldsymbol{y}_A)$ between f and a set of observations $\boldsymbol{y}_A$, where these observations correspond to a set $A = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots\}$ and each $\boldsymbol{x}_i$ in A is also in D. For a GP, $I(f; \boldsymbol{y}_A)$ is

$$I(f; \boldsymbol{y}_A) = H(\boldsymbol{y}_A) - H(\boldsymbol{y}_A \,|\, f) = \frac{1}{2} \log \left| \mathbf{I} + \sigma_n^{-2} K(A, A) \right|,$$

where $K(A, A)$ is the covariance matrix of the values of f at the elements of the set A, $H(\boldsymbol{y}_A)$ is the differential entropy of the probability distribution over the set of observations $\boldsymbol{y}_A$, and $H(\boldsymbol{y}_A \,|\, f)$ is the corresponding value when the distribution over $\boldsymbol{y}_A$ is conditioned on f. Note that for a GP, since $\boldsymbol{y}_A$ only depends on the values of f at A, denoted $f(A)$, it follows that $H(\boldsymbol{y}_A \,|\, f) = H(\boldsymbol{y}_A \,|\, f(A))$ and so $I(f; \boldsymbol{y}_A) = I(f(A); \boldsymbol{y}_A)$.

The *conditional mutual information* with respect to f resulting from observations $\boldsymbol{y}_A$, given previous observations $\boldsymbol{y}_S$, is defined (for two finite sets $A, S \subseteq D$) as

$$I(f; \boldsymbol{y}_A \mid \boldsymbol{y}_S) = H(\boldsymbol{y}_A \mid \boldsymbol{y}_S) - H(\boldsymbol{y}_A \mid f, \boldsymbol{y}_S) = H(\boldsymbol{y}_A \mid \boldsymbol{y}_S) - H(\boldsymbol{y}_A \mid f),$$

where the second equality follows from conditional independence of the observations given f. The conditional mutual information gained from observations $\boldsymbol{y}_A$ can also be calculated as a sum of the marginal conditional mutual information gains of each observation in $\boldsymbol{y}_A$; conditioned on $\boldsymbol{y}_S$, and for $A = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$, this sum is

$$I(f; \boldsymbol{y}_A \mid \boldsymbol{y}_S) = \sum_{t=1}^{T} \log\left(1 + \sigma_n^{-2}\sigma^2_{t-1}(\boldsymbol{x}_t)\right), \tag{3}$$

where the term $\sigma^2_{t-1}(\boldsymbol{x}_t)$ is the posterior variance over $f(\boldsymbol{x}_t)$, conditioned on $\boldsymbol{y}_S$ and $\{y_1, ..., y_{t-1}\} \subseteq \boldsymbol{y}_A$. It is important to note that $\sigma^2_{t-1}(\boldsymbol{x}_t)$, given by Equation (2), is independent of the values of the observations. Since the sum's value can thus be calculated without making the observations (i.e., during the course of assembling a batch), it is possible to calculate the mutual information that will be gained from any hypothetical set of

Algorithm 1 GP-UCB

Input: Decision set D, GP prior μ_0, σ_0, kernel function $k(\cdot,\cdot)$.
for $t = 1, 2, \ldots, T$ **do**
  Choose $\boldsymbol{x}_t = \text{argmax}_{\boldsymbol{x}\in D}[\mu_{t-1}(\boldsymbol{x}) + \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x})]$
  Compute $\sigma_t(\cdot)$ via Equation (2)
  Obtain $y_t = f(\boldsymbol{x}_t) + \varepsilon_t$
  Perform Bayesian inference to obtain $\mu_t(\cdot)$ via Equation (1)
end for

observations. We will also be interested in the *maximum* information gain with respect to f obtainable from observations $\boldsymbol{y}_A$ corresponding to any set of actions A, where $|A| \leq T$,

$$\gamma_T = \max_{A\subseteq D,\, |A|\leq T} I(f; \mathbf{y}_A). \tag{4}$$

3.4 The GP-UCB Approach for Strictly Sequential Selection

Modeling f as a sample from a GP has the major benefit that the predictive uncertainty can be used to guide exploration and exploitation. This is done via a decision rule, by which the algorithm selects actions $\boldsymbol{x}_t$. While several heuristics, such as Expected Improvement (Mockus et al., 1978) and Most Probable Improvement (Mockus, 1989) have been effectively employed in practice, nothing is known about their convergence properties in the case of noisy observations. Srinivas et al. (2010), guided by the success of upper-confidence-based sampling approaches for multi-armed bandit problems (Auer, 2002), analyzed the Gaussian process Upper Confidence Bound (GP-UCB) decision rule,

$$\boldsymbol{x}_t = \operatorname*{argmax}_{\boldsymbol{x}\in D} \left[\mu_{t-1}(\boldsymbol{x}) + \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x})\right]. \tag{5}$$

This decision rule uses α_t, a domain-specific time-varying parameter, to trade off exploitation (sampling $\boldsymbol{x}$ with high mean) and exploration (sampling $\boldsymbol{x}$ with high standard deviation). Srinivas et al. (2010) showed that, with proper choice of α_t, the cumulative regret of GP-UCB grows sublinearly for many commonly used kernel functions. This algorithm is presented in simplified pseudocode as Algorithm 1.

Implicit in the definition of the GP-UCB decision rule is the corresponding confidence interval for each $\boldsymbol{x} \in D$,

$$C_t^{\text{seq}}(\boldsymbol{x}) \equiv \left[\mu_{t-1}(\boldsymbol{x}) - \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x}),\ \mu_{t-1}(\boldsymbol{x}) + \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x})\right], \tag{6}$$

where this confidence interval's upper confidence bound is the value of the argument of the decision rule. For this (or any) confidence interval, we will refer to the difference between the uppermost limit and the lowermost, here $w = 2\alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x})$, as the *width*. This confidence interval is based on the posterior over f given $\boldsymbol{y}_{1:t-1}$; a new confidence interval is created for round $t+1$ after adding y_t to the set of observations. Srinivas et al. (2010) carefully select α_t such that a union bound over all $t \geq 1$ and $\boldsymbol{x} \in D$ yields a high-probability guarantee of confidence interval correctness; it is this guarantee and the direct relationship

between confidence intervals and the decision rule which enable the construction of high-probability regret bounds. Using this guarantee, Srinivas et al. (2010) then prove that the cumulative regret of the GP-UCB algorithm can be bounded as $R_T = O(\sqrt{T\alpha_T\gamma_T})$, where α_T is the confidence interval width multiplier described above. For many commonly used kernel functions, Srinivas et al. (2010) show that γ_T grows sublinearly and α_T only needs to grow polylogarithmically in T, implying that R_T is also sublinear; thus $R_T/T \to 0$ as $T \to \infty$, i.e., GP-UCB is a no-regret algorithm.

Motivated by the strong theoretical and empirical performance of GP-UCB (Srinivas et al., 2010, 2012), we explore generalizations to batch and parallel selection (i.e., $B > 1$). One naïve approach would be to update the GP-UCB score, Equation (5), only once new feedback becomes available, selecting the same action at each time step between acquisitions of new observations. In the case that the observation noise model is Gaussian, the bound of Srinivas et al. (2010) can be used together with reparameterization of time to bound the regret to no more than a factor of $\sqrt{B}$ greater than the GP-UCB algorithm. In empirical tests (Online Appendix 2), this algorithm does not explore sufficiently to perform well early on, making it of limited practical interest. To encourage more exploration, one may instead require that no action is selected twice within a batch (i.e., simply rank actions according to the GP-UCB score, and pick actions in order of decreasing score until new feedback is available). However, since f often varies smoothly, so does the GP-UCB score; under some circumstances, this algorithm would also suffer from limited exploration. Further, if the optimal set $\boldsymbol{X}^* \subseteq D$ is of size $|\boldsymbol{X}^*| < B$ and there is a finite gap between the rewards $f(\boldsymbol{x}^*)$ and $f(\boldsymbol{x})$ for all $\boldsymbol{x}^* \in \boldsymbol{X}^*, \boldsymbol{x} \notin \boldsymbol{X}^*$, the algorithm is suffers linear regret, since some $\boldsymbol{x} \notin \boldsymbol{X}^*$ must be included in every batch. This algorithm also underperforms in empirical tests (Online Appendix 2). These naïve algorithms have clear shortcomings because they do not simultaneously select diverse sets of actions and ensure appropriate convergence behavior.

In the following, we introduce the *Gaussian process Batch Upper Confidence Bound (GP-BUCB)* algorithm, which successfully balances these competing imperatives. GP-BUCB encourages diversity in exploration, uses past information in a principled fashion, and yields strong performance guarantees. We also extend it and develop the *Gaussian process Adaptive Upper Confidence Bound (GP-AUCB)* algorithm, which retains the theoretical guarantees of the GP-BUCB algorithm, but chooses batches of variable length in an adaptive, data-driven manner.

4. The GP-BUCB Algorithm and Regret Bounds

We introduce the GP-BUCB algorithm in Section 4.1. Section 4.2 states the paper's major theorem, a bound on the cumulative regret of a general class of algorithms including GP-BUCB and GP-AUCB. This main result is in terms of a quantity C, a bound on information used within a batch; this quantity is examined in detail in Section 4.3. Using these insights, Section 4.4 provides a corollary, bounding the regret of GP-BUCB specifically. Section 4.5 improves this regret bound by initializing GP-BUCB with a finite set of observations.

4.1 GP-BUCB: An Overview

A key property of GPs is that the predictive variance at time t, Equation (2), only depends on $\boldsymbol{X}_{t-1} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}\}$, i.e., *where* the observations are made, but not *which*

Algorithm 2 GP-BUCB

Input: Decision set D, GP prior μ_0, σ_0, kernel function $k(\cdot,\cdot)$, feedback mapping $\mathrm{fb}[\cdot]$.
for $t = 1, 2, \ldots, T$ **do**
  Choose $\boldsymbol{x}_t = \mathrm{argmax}_{\boldsymbol{x}\in D}[\mu_{\mathrm{fb}[t]}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})]$
  Compute $\sigma_t(\cdot)$ via Equation (2)
  if $\mathrm{fb}[t] < \mathrm{fb}[t+1]$ **then**
    Obtain $y_{t'} = f(\boldsymbol{x}_{t'}) + \varepsilon_{t'}$ for $t' \in \{\mathrm{fb}[t]+1, \ldots, \mathrm{fb}[t+1]\}$
    Perform Bayesian inference to obtain $\mu_{\mathrm{fb}[t+1]}(\cdot)$ via Equation (1)
  end if
end for

values $\boldsymbol{y}_{1:t-1} = [y_1, \ldots, y_{t-1}]^T$ were actually observed. Thus, it is possible to compute the posterior variance that would be used by the sequential GP-UCB decision rule, Equation (5), even while certain observations are not yet available. In contrast, the predictive mean using in Equation (1) does depend on the actual observations. A natural approach towards parallel exploration is therefore to replace the GP-UCB decision rule, Equation (5), with a decision rule that sequentially chooses actions within the batch using all the information that is available so far,

$$\boldsymbol{x}_t = \operatorname*{argmax}_{\boldsymbol{x}\in D}\left[\mu_{\mathrm{fb}[t]}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})\right]. \tag{7}$$

Here, the parameter β_t has a role analogous to the parameter α_t in the GP-UCB algorithm. The confidence intervals corresponding to this decision rule are of the form

$$C_t^{\mathrm{batch}}(\boldsymbol{x}) \equiv \left[\mu_{\mathrm{fb}[t]}(\boldsymbol{x}) - \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}),\ \mu_{\mathrm{fb}[t]}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})\right]. \tag{8}$$

Note that this approach is equivalent to running the strictly sequential GP-UCB algorithm based on *hallucinated observations*. Concretely, we *hallucinate* observations $\boldsymbol{y}_{\mathrm{fb}[t]+1:t-1}$ for those observations that have not yet been received, simply using the most recently updated posterior mean, i.e., $\boldsymbol{y}_{\mathrm{fb}[t]+1:t-1} = [\mu_{\mathrm{fb}[t]}(\boldsymbol{x}_{\mathrm{fb}[t]+1}), \ldots, \mu_{\mathrm{fb}[t]}(\boldsymbol{x}_{t-1})]$. As a consequence, the mean of the posterior including these hallucinated observations remains precisely $\mu_{\mathrm{fb}[t]}(\boldsymbol{x})$, but the posterior variance decreases.

The resulting GP-BUCB algorithm is shown in pseudocode as Algorithm 2. This approach naturally encourages diversity in exploration by taking into account the change in predictive variance that will eventually occur after receiving the pending observations; since the payoffs of "similar" actions are assumed to co-vary, exploring one action will automatically reduce the predictive variance of similar actions, and thus their value in terms of exploration. This decision rule appropriately deprecates those observations which will be made partially redundant by the acquisition of the pending observations, resulting in a more correct valuation of exploring any $\boldsymbol{x}$ in D.

The disadvantage of this approach appears as the algorithm progresses deeper into the batch. At each time t, the width of the confidence intervals $C_t^{\mathrm{batch}}(\boldsymbol{x})$ is proportional to $\sigma_{t-1}(\boldsymbol{x})$. As desired, shrinking the confidence intervals with respect to the start of the batch by using this standard deviation enables GP-BUCB to avoid exploratory redundancy. However, as an undesired side-effect, doing so conflates the information which is actually available, gained via the observations $\boldsymbol{y}_{1:\mathrm{fb}[t]}$, with the hallucinated information corresponding

to actions $\boldsymbol{x}_{\text{fb}[t]+1}$ through $\boldsymbol{x}_{t-1}$. Thus, the posterior reflected by $\sigma_{t-1}(\boldsymbol{x})$ is "overconfident" about the algorithm's actual state of knowledge of the function. This is problematic when using the confidence intervals to bound the regret.

To build an algorithm with rigorous guarantees on its performance while still avoiding exploratory redundancy, we must control for this overconfidence. One measure of overconfidence is the ratio $\sigma_{\text{fb}[t]}(\boldsymbol{x})/\sigma_{t-1}(\boldsymbol{x})$, which is the ratio of the width of the confidence interval derived from the set of actual observations $\boldsymbol{y}_{1:\text{fb}[t]}$ to the width of the confidence interval derived from the partially hallucinated set of observations $\boldsymbol{y}_{1:t-1}$. This ratio is related to $I(f(\boldsymbol{x}); \boldsymbol{y}_{\text{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\text{fb}[t]})$, the hallucinated conditional mutual information with respect to $f(\boldsymbol{x})$ (as opposed to the whole of f), as follows:

Proposition 1 *The ratio of the standard deviation of the posterior over $f(\boldsymbol{x})$, conditioned on observations $\boldsymbol{y}_{1:fb[t]}$, to that conditioned on $\boldsymbol{y}_{1:fb[t]}$ and hallucinated observations $\boldsymbol{y}_{fb[t]+1:t-1}$ is*

$$\frac{\sigma_{fb[t]}(\boldsymbol{x})}{\sigma_{t-1}(\boldsymbol{x})} = \exp\Big(I(f(\boldsymbol{x}); \boldsymbol{y}_{fb[t]+1:t-1} \mid \boldsymbol{y}_{1:fb[t]})\Big).$$

Proof The proposition follows from the fact that

$$\begin{aligned} I(f(\boldsymbol{x}); \boldsymbol{y}_{\text{fb}[t]+1:t-1} | \boldsymbol{y}_{1:\text{fb}[t]}) &= H(f(\boldsymbol{x}) | \boldsymbol{y}_{1:\text{fb}[t]}) - H(f(\boldsymbol{x}) | \boldsymbol{y}_{1:t-1}) \\ &= 1/2 \log(2\pi e \sigma^2_{\text{fb}[t]}(\boldsymbol{x})) - 1/2 \log(2\pi e \sigma^2_{t-1}(\boldsymbol{x})) \\ &= \log(\sigma_{\text{fb}[t]}(\boldsymbol{x})/\sigma_{t-1}(\boldsymbol{x})). \end{aligned}$$

■

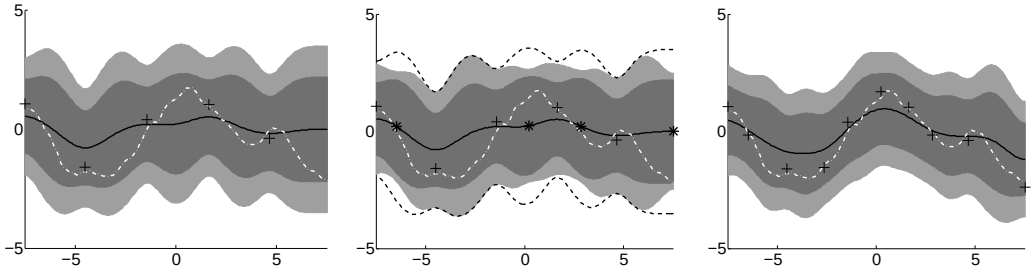
Crucially, if there exists some constant C, such that $I(f(\boldsymbol{x}); \boldsymbol{y}_{\text{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\text{fb}[t]}) \leq C$, $\forall \boldsymbol{x} \in D, \forall t \geq 1$, the ratio $\sigma_{\text{fb}[t]}(\boldsymbol{x})/\sigma_{t-1}(\boldsymbol{x})$ can also be bounded for every $\boldsymbol{x} \in D$ as follows:

$$\frac{\sigma_{\text{fb}[t]}(\boldsymbol{x})}{\sigma_{t-1}(\boldsymbol{x})} = \exp\Big(I(f(\boldsymbol{x}); \boldsymbol{y}_{\text{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\text{fb}[t]})\Big) \leq \exp(C). \tag{9}$$

Armed with such a bound, the algorithm can be modified to compensate for its overconfidence. Our goal is to compensate in a way that allows the algorithm to avoid redundancy, while guaranteeing accurate confidence intervals for the sake of deriving regret bounds. Our strategy is to increase the width of the confidence intervals (through proper choice of the parameter β_t), such that the confidence intervals used by GP-BUCB are conservative in their use of the hallucinated information and consequently still contain the payoff function $f(\boldsymbol{x})$ with high probability. More precisely, we will require that $C^{\text{seq}}_{\text{fb}[t]+1}(\boldsymbol{x}) \subseteq C^{\text{batch}}_t(\boldsymbol{x})$ for all t at which we select actions and all $\boldsymbol{x} \in D$; that is, the batch algorithm's confidence intervals are sufficiently large to guarantee that even for the last action selection in the batch, they contain the confidence intervals used by the GP-UCB algorithm given $\boldsymbol{y}_{1:\text{fb}[t]}$, as defined in Equation (6). Srinivas et al. (2010) provide choices of α_t such that the resulting confidence intervals have a high-probability guarantee of correctness $\forall t \geq 1, \boldsymbol{x} \in D$. Thus, if it can be shown that $C^{\text{seq}}_{\text{fb}[t]+1}(\boldsymbol{x}) \subseteq C^{\text{batch}}_t(\boldsymbol{x})$, $\forall \boldsymbol{x} \in D$, $t \in \mathbb{N}$, the batch confidence intervals inherit the high-probability guarantee of correctness.

Fortunately, the relationship between $C^{\text{seq}}_{\text{fb}[t]+1}(\boldsymbol{x})$ and $C^{\text{batch}}_t(\boldsymbol{x})$ is simple; since the partially hallucinated posterior has the same mean as that based on only $\boldsymbol{y}_{1:\text{fb}[t]}$,

$$C^{\text{seq}}_{\text{fb}[t]+1}(\boldsymbol{x}) \subseteq C^{\text{batch}}_t(\boldsymbol{x}) \iff \beta_t^{1/2} \sigma_{t-1}(\boldsymbol{x}) \geq \alpha^{1/2}_{\text{fb}[t]} \sigma_{\text{fb}[t]}(\boldsymbol{x}).$$



(a) *Beginning of batch 2* (b) *Last decision of batch 2* (c) *Beginning of batch 3*

Figure 1: (a): The confidence intervals $C^{\mathrm{seq}}_{\mathrm{fb}[t]+1}(\boldsymbol{x})$ (dark), computed from previous noisy observations $\boldsymbol{y}_{1:\mathrm{fb}[t]}$ (crosses), are centered around the posterior mean (solid black) and contain $f(\boldsymbol{x})$ (white dashed) w.h.p. To avoid overconfidence, GP-BUCB chooses $C^{\mathrm{batch}}_{\mathrm{fb}[t]+1}(\boldsymbol{x})$ (light gray) such that even in the worst case, the succeeding confidence intervals in the batch, $C^{\mathrm{batch}}_{\tau}(\boldsymbol{x})$, $\forall \tau : \mathrm{fb}[\tau] = \mathrm{fb}[t]$, will contain $C^{\mathrm{seq}}_{\mathrm{fb}[t]+1}(\boldsymbol{x})$. (b): Due to the observations that GP-BUCB "hallucinates" (stars), the outer posterior confidence intervals $C^{\mathrm{batch}}_{t}(\boldsymbol{x})$ shrink from their values at the start of the batch (black dashed), but still contain $C^{\mathrm{seq}}_{\mathrm{fb}[t]+1}(\boldsymbol{x})$, as desired. (c): Upon selection of the last action of the batch, the feedback for all actions is obtained, and for the subsequent action selection in round t', new confidence intervals $C^{\mathrm{seq}}_{\mathrm{fb}[t']+1}(\boldsymbol{x})$ and $C^{\mathrm{batch}}_{\mathrm{fb}[t']+1}(\boldsymbol{x})$ are computed.

If we have a suitable bound on $\sigma_{\mathrm{fb}[t]}(\boldsymbol{x})/\sigma_{t-1}(\boldsymbol{x})$ via Equation (9), all that remains is to choose β_t appropriately. If we do so by using a uniform, multiplicative increase with respect to $\alpha_{\mathrm{fb}[t]}$ for every $\boldsymbol{x} \in D$ and $t \in \mathbb{N}$, the desired redundancy avoidance property of these confidence intervals is simultaneously maintained, since the actions corresponding to pending observations (and related actions) are deprecated as if the observations had actually been obtained. Figure 1 illustrates this idea. The problem of developing a parallel algorithm with bounded delay is thus reduced to finding a value C such that $I(f(\boldsymbol{x}); \boldsymbol{y}_{\mathrm{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\mathrm{fb}[t]}) \leq C$, $\forall \boldsymbol{x} \in D, \forall t \geq 1$, thus allowing us to select β_t to guarantee the containment of the reference sequential confidence intervals by their batch counterparts.

4.2 General Regret Bound

Our main theorem bounds the regret of GP-BUCB and related algorithms. This regret bound is formulated in terms of a bound C, which we assume to be known to the algorithm, on the maximum amount of conditional mutual information which is hallucinated with respect to $f(\boldsymbol{x})$ for any $\boldsymbol{x}$ in D. We defer discussion of methods of obtaining such a bound to Section 4.3. This bound is used to relate confidence intervals used to select actions, which incorporate this hallucinated information, to the posterior confidence intervals as of the last feedback obtained, which contain the payoff function f with high probability. This theorem holds under any of three different assumptions about f, studied by Srinivas et al. (2012) in the case of the GP-UCB algorithm, which may all be of practical interest. In particular, it

holds even if the assumption that f is sampled from a GP is replaced by the assumption that f has low norm in the associated Reproducing Kernel Hilbert Space (RKHS).[3]

Theorem 2 *Specify $\delta \in (0,1)$ and let γ_t be as defined in Equation (4). Let there exist a mapping $fb[t]$ (possibly revealed online) that dictates at which rounds new feedback becomes available. Model the payoff function f via a Gaussian process prior with bounded variance, such that for any $\boldsymbol{x}$ in the decision set D, $k(\boldsymbol{x},\boldsymbol{x}) \leq 1$. Suppose one of the following sets of assumptions holds:*

Case 1: D is a finite set and f is sampled from the assumed GP prior. The noise variables ϵ_t are i.i.d., $\epsilon_t \sim \mathcal{N}(0,\sigma_n^2)$. Choose $\alpha_t = 2\log(|D|t^2\pi^2/6\delta)$.

Case 2: $D \subseteq [0,l]^d$ is compact and convex, with $d \in \mathbb{N}$, $l > 0$, and f is sampled from the assumed zero-mean GP prior. The noise variables ϵ_t are i.i.d., $\epsilon_t \sim \mathcal{N}(0,\sigma_n^2)$. The kernel $k(\boldsymbol{x},\boldsymbol{x}')$ is such that the following bound holds with high probability on the derivatives of GP sample paths f, where a and b are constants such that $a \geq \delta/(4d)$, $b > 0$ and $bl\sqrt{\log(4da/\delta)}$ is an integer:

$$Pr\left\{\sup_{\boldsymbol{x}\in D} |\partial f/\partial x_j| > L\right\} \leq ae^{-(L/b)^2},\ j = 1,\ldots,d.$$

Choose $\alpha_t = 2\log(2t^2\pi^2/(3\delta)) + 2d\log\left(t^2dbl\sqrt{\log(4da/\delta)}\right)$.

Case 3: D is arbitrary and the squared RKHS norm of f under the kernel assumed is bounded as $||f||_k^2 \leq M$ for some constant M. The noise variables ε_t form an arbitrary *martingale difference sequence (meaning that $\mathbb{E}[\varepsilon_t \,|\, \varepsilon_1,\ldots,\varepsilon_{t-1}] = 0$ for all $t \in \mathbb{N}$), uniformly bounded by σ_n. Choose $\alpha_t = 2M + 300\gamma_t \ln^3(t/\delta)$.*

Employ the GP posterior and the GP-BUCB update rule, Equation (7), to select actions $\boldsymbol{x}_t \in D$ for all $t \geq 1$, using $\beta_t = \exp(2C)\alpha_{fb[t]+1}$ (Cases 1 & 3) or $\beta_t = \exp(2C)\alpha_t$ (Case 2), where $C > 0$ and

$$I(f(\boldsymbol{x}); \boldsymbol{y}_{fb[t]+1:t-1} \mid \boldsymbol{y}_{1:fb[t]}) \leq C, \tag{10}$$

for all $t \geq 1$ and all $\boldsymbol{x} \in D$. Under these conditions, the following statement holds with regard to the cumulative regret:

$$\Pr\left\{R_T \leq \sqrt{C_1 T \exp(2C)\alpha_T\gamma_T} + 2, \forall\, T \geq 1\right\} \geq 1-\delta,$$

where $C_1 = 8/\log(1+\sigma_n^{-2})$.

Proof The proof of this result is presented in Appendix A. ■

First, note that this guarantee holds for any amount of time the algorithm is allowed to run, since the algorithm does not use knowledge of how many actions it may yet take; thus, with high probability, R_T is less than the given expression for every T less than or equal to the number of executed actions. Second, the key quantity that controls the regret in Theorem 2 is C, the bound in Equation (10) on the maximum conditional mutual information obtainable within a batch with respect to $f(\boldsymbol{x})$ for any $\boldsymbol{x} \in D$. In particular, the cumulative regret bound of Theorem 2 is a factor $\exp(C)$ larger than the regret bound for the sequential ($B = 1$) GP-UCB algorithm. Various choices of the key parameter C are explored in the following sections.

3. See Schölkopf and Smola (2002).

4.3 Suitable Choices for C

The significance of a bound C on the information hallucinated with respect to any $f(\boldsymbol{x})$ arises through this quantity's ability to bound the degree of contamination of the GP-BUCB confidence intervals, given by Equation (8), with hallucinated information.

Two properties of the mutual information in this setting are particularly useful. These properties are monotonicity (adding an element $\boldsymbol{x}$ to the set A cannot decrease the mutual information between f and the corresponding set of observations $\boldsymbol{y}_A$) and submodularity (the increase in mutual information between f and $\boldsymbol{y}_A$ with the addition of an element $\boldsymbol{x}$ to set A cannot be greater than the corresponding increase in mutual information if $\boldsymbol{x}$ is added to A', where $A' \subseteq A$) (Krause and Guestrin, 2005). Submodularity arises because individual observations are conditionally independent, given f.

Using the time indexing notation developed in Section 3.1, the following results hold:

$$\forall \boldsymbol{x} \in D: \quad I(f(\boldsymbol{x}); \boldsymbol{y}_{\mathrm{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\mathrm{fb}[t]}) \leq I(f; \boldsymbol{y}_{\mathrm{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\mathrm{fb}[t]}) \tag{11}$$

$$\leq \max_{A \subseteq D, |A| \leq B-1} I(f; \boldsymbol{y}_A \mid \boldsymbol{y}_{1:\mathrm{fb}[t]}) \tag{12}$$

$$\leq \max_{A \subseteq D, |A| \leq B-1} I(f; \boldsymbol{y}_A) = \gamma_{B-1}. \tag{13}$$

The first inequality follows from the monotonicity of mutual information, i.e., the information gained with respect to f as a whole must be at least as large as that gained with respect to $f(x)$. The second inequality holds because we specify the feedback mapping such that $t - \mathrm{fb}[t] \leq B$, and the third inequality holds due to the submodularity of the conditional mutual information.

Often, the terms on the right-hand side of these inequalities are easier to work with than $I(f(\boldsymbol{x}); \boldsymbol{y}_{\mathrm{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\mathrm{fb}[t]})$. The remainder of the paper is characterized by which inequality we employ in constructing an algorithm and choosing a suitable C to use with Equation (9) and Theorem 2; Sections 4.4 and 4.5 approach the problem via Inequalities (13) and (12), while Section 5.1 exploits Inequality (11) and Section 5.2 examines the consequences of directly bounding the local hallucinated information.

4.4 Corollary Regret Bound: GP-BUCB

The GP-BUCB algorithm requires that $t - \mathrm{fb}[t] \leq B,\ \forall t \geq 1$, and uses a value C such that, for any $t \in \mathbb{N}$,

$$\max_{A \subseteq D, |A| \leq B-1} I(f; \boldsymbol{y}_A \mid \boldsymbol{y}_{1:\mathrm{fb}[t]}) \leq C, \tag{14}$$

thus bounding $I(f(\boldsymbol{x}); \boldsymbol{y}_{\mathrm{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\mathrm{fb}[t]})$ for all $\boldsymbol{x} \in D$ and $t \in \mathbb{N}$ via Inequality (12). Otherwise stated, in GP-BUCB, the local information gain with respect to any $f(\boldsymbol{x}), \boldsymbol{x} \in D, t \in \mathbb{N}$ is bounded by fixing the feedback times and then bounding the maximum conditional mutual information with respect to the entire function f which can be acquired by *any* algorithm which chooses any set of $B-1$ or fewer observations. This approach is sensible because such a bound C holds for any batches constructed with any algorithm. Following an approach which is less agnostic with regard to algorithm choice makes it quite difficult to disentangle the role of C in setting the exploration-exploitation tradeoff parameter β_t from its role as a bound on how much information is hallucinated by the algorithm; since a larger β_t encourages exploration under the GP-BUCB decision rule, Equation (7), a larger value of C

Algorithm 3 Uncertainty Sampling

Input: Decision set D, GP prior μ_0, σ_0, kernel function $k(\cdot,\cdot)$.
for $t = 1, 2, \ldots, T$ **do**
  Choose $\boldsymbol{x}_t = \text{argmax}_{\boldsymbol{x} \in D} \sigma_{t-1}(\boldsymbol{x})$
  Compute $\sigma_t(\cdot)$ via Equation (2)
end for

(and thus β_t) typically produces batches that explore more and thus use more hallucinated information.

It remains to choose a C which satisfies Inequality (14). We do so via Inequality (13). As noted in Section 4.3, mutual information is submodular with respect to the set of observed actions, and thus the maximum conditional mutual information which can be gained by making any set of observations is maximized when the set of observations currently available, to which these new observations will be added, is empty. Letting the maximum mutual information between f and any observation set of size $B-1$ be denoted γ_{B-1} and choosing $C = \gamma_{B-1}$ provides a bound on the possible local conditional mutual information gain for any $t \in \mathbb{N}$ and $\boldsymbol{x} \in D$, as in Inequality (13).

In practice, γ_{B-1} is often difficult to calculate; in general, this requires optimizing over the combinatorially large set of sets of actions of size $B-1$. However, Krause and Guestrin (2005) demonstrate that, due to the submodularity of the mutual information with respect to f in this setting, there is an easily obtained upper bound on γ_{B-1}. Specifically, they use uncertainty sampling, a greedy procedure, shown here as Algorithm 3, and show that $e/(e-1)\, I(f; \boldsymbol{y}^{US}_{B-1}) \geq \gamma_{B-1}$, where $I(f; \boldsymbol{y}^{US}_{B-1})$ is the information gained by observing the set of observations $\boldsymbol{y}^{US}_{B-1}$ corresponding to the actions $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{B-1}\}$ selected using uncertainty sampling. This insight enables efficient computation of upper bounds on γ_{B-1}.

Choosing $C = \gamma_{B-1}$ yields the following Corollary, a special case of Theorem 2:

Corollary 3 *Assume the GP-BUCB algorithm is employed with a constant B such that $t - fb[t] \leq B$ for all $t \geq 1$. Let $\delta \in (0,1)$, and let the requirements of one of the numbered cases of Theorem 2 be met. Choose $\beta_t = \exp(2C)\alpha_{fb[t]+1}$ (Cases 1 & 3) or $\beta_t = \exp(2C)\alpha_t$ (Case 2) and select actions $\boldsymbol{x}_t$ for all $t \geq 1$. Then*

$$\Pr\left\{R_T \leq \sqrt{C_1 T \exp(2\gamma_{B-1})\alpha_T \gamma_T} + 2, \quad \forall T \geq 1\right\} \geq 1 - \delta,$$

where $C_1 = 8/\log(1+\sigma_n^{-2})$ and γ_{B-1} and γ_τ are as defined in Equation (4).

Unfortunately, the choice $C = \gamma_{B-1}$ is not especially satisfying from the perspective of asymptotic scaling. The maximum information gain γ_{B-1} usually grows at least as $\Omega(\log B)$, implying that $\exp(C)$ grows at least linearly in B, yielding a regret bound which is also at least linear in B. Fortunately, the analysis of Section 4.5 shows that the GP-BUCB algorithm can be modified such that a constant choice of C *independent* of B suffices.

4.5 Better Bounds Through Initialization

To obtain regret bounds *independent* of batch size B, the monotonicity properties of conditional mutual information can again be exploited. This can be done by structuring GP-BUCB as a two-stage procedure. First, an *initialization set* D^{init} of size $|D^{\text{init}}| = T^{\text{init}}$ is

selected nonadaptively (i.e., without any feedback); following the selection of this entire set, feedback $\boldsymbol{y}_{D^{\text{init}}}$ for all actions in $D^{\text{init}} = \{\boldsymbol{x}_1^{\text{init}}, \ldots, \boldsymbol{x}_{T^{\text{init}}}^{\text{init}}\}$ is obtained. In the second stage, GP-BUCB is applied to the posterior Gaussian process distribution, conditioned on $\boldsymbol{y}_{D^{\text{init}}}$.

Notice that if we define

$$\gamma_T^{\text{init}} = \max_{A \subseteq D, |A| \leq T} I(f; \boldsymbol{y}_A \mid \boldsymbol{y}_{D^{\text{init}}}),$$

then, under the assumptions of Theorem 2, using $C = \gamma_{B-1}^{\text{init}}$, the regret of the two-stage algorithm is bounded by $R_T = O\left(T^{\text{init}} + (T\gamma_T \alpha_T \exp(2C))^{1/2}\right)$. In the following, we show that it is indeed possible to construct an initialization set D^{init} such that the size T^{init} is dominated by $(T\gamma_T\alpha_T \exp(2C))^{1/2}$, and—crucially—that $C = \gamma_{B-1}^{\text{init}}$ can be bounded *independently* of the batch size B.

The initialization set D^{init} which enables us to make this argument is constructed by running the uncertainty sampling algorithm (Algorithm 3) for T^{init} rounds and setting D^{init} to the selected actions. Note that uncertainty sampling can be viewed as a special case of the GP-BUCB algorithm with a constant prior mean of 0 and the requirement that for all $1 \leq t \leq T^{\text{init}}$, $\text{fb}[t] = 0$, i.e., no feedback is taken into account for the first T^{init} iterations.

Under this procedure, we have the following key result about the maximum residual information gain γ^{init}:

Lemma 4 *Suppose uncertainty sampling is used to generate an initialization set D^{init} of size T^{init}. Then*

$$\gamma_{B-1}^{init} \leq \frac{B-1}{T^{init}} \gamma_{T^{init}}. \tag{15}$$

Proof The proof of this lemma is presented in Appendix B. ■

Whenever γ_T is sublinear in T, the bound on $\gamma_{B-1}^{\text{init}}$ given by Inequality (15) converges to zero for sufficiently large T^{init}; thus for *any* constant $C > 0$, we can choose T^{init} as a function of B such that $\gamma_{B-1}^{\text{init}} < C$. Using this choice of C in Theorem 2 bounds the post-initialization regret. In order to derive bounds on T^{init}, we in turn need a bound on γ_T which is analytical and sublinear. Fortunately, Srinivas et al. (2010) prove suitable bounds on how the information gain γ_T grows for some of the most commonly used kernels. We summarize our analysis below in Theorem 5. For sake of notation, define R_T^{seq} to be the regret bound of Corollary 3 with $B = 1$ (i.e., that of Srinivas et al., 2010, associated with the sequential GP-UCB algorithm).

Theorem 5 *Suppose the assumptions of one of the cases of Theorem 2 are satisfied. Further, suppose the kernel and T^{init} are as listed in Table 1, and $B \geq 2$. Fix $\delta > 0$. Let R_T be the cumulative regret at round T of the two-stage initialized GP-BUCB algorithm, which ignores feedback for the first T^{init} rounds. Then there exists a constant C' independent of B such that*

$$Pr\left\{R_T \leq C' R_T^{seq} + 2||f||_\infty T^{init}, \forall T \geq 1\right\} \geq 1 - \delta, \tag{16}$$

where C' takes the value shown in Table 1.

Proof The proof of this result and the values in Table 1 are presented in Appendix B. ■

In Table 1, $\lceil \cdot \rceil$ denotes the first integer greater than or equal to the argument. Note that

Kernel Type	**Size T^{init} of Initialization Set D^{init}**	**Regret Multiplier C'**
Linear: $\gamma_t \leq \eta d \log(t+1)$	$\left\lceil \max \left[\log(B), \frac{\log\eta + \log d + 2\log(B)}{2\log(B)-1} e\eta d(B-1)\log(B) \right] \right\rceil$	$\exp(2/e)$
Matérn: $\gamma_t \leq \nu t^{\epsilon}$	$\lceil (\nu(B-1))^{1/(1-\epsilon)} \rceil$	e
RBF: $\gamma_t \leq \eta(\log(t+1))^{d+1}$	$\left\lceil \max \left[(\log(B))^{d+1}, \left(\frac{e}{d+1}\frac{\log\eta+(d+2)\log(B)}{2\log(B)-1}\right)^{d+1} \eta(B-1)(\log(B))^{d+1} \right] \right\rceil$	$\exp((\frac{2d+2}{e})^{d+1})$

Table 1: Initialization set sizes for Theorem 5.

the particular values of C' used in Table 1 are not the only ones possible; they are chosen simply because they yield relatively clean algebraic forms for T^{init}. The most important component of this result is the scaling of the regret R_T with T and B. As compared to Theorem 2, which bounds R_T via the product $\exp(2C)T\alpha_T\gamma_T$, where C is a function of B, Theorem 5 replaces the root of this product with a sum of two terms, one in each of B and T; the term $C'R_T^{\text{seq}}$ in Inequality (16) is the cost paid for running the algorithm post-initialization (dependent on T, but not B), whereas the second term is the cost of performing the initialization (dependent on B, but not T). Notice that whenever $B = O(\text{polylog}(T))$, $T^{\text{init}} = O(\text{polylog}(T))$, and further, note $R_T^{\text{seq}} = \Omega(\sqrt{T})$. Thus, as long as the batch size does not grow too quickly, the term $O(T^{\text{init}})$ is dominated by $C'R_T^{\text{seq}}$ and the regret bounds of GP-BUCB are only a constant factor, *independent of* B, worse than those of GP-UCB.

In practice, D^{init} should not be constructed by running uncertainty sampling for T^{init} rounds, but rather by running until $\gamma_{B-1}^{\text{init}} \leq C$ for the pre-specified C; one online check can be constructed using Lemma 4. This procedure cannot take more than T^{init} rounds for the kernels discussed and may take considerably fewer. Further, this procedure is applicable to any kernel with sublinear γ_T, generalizing this initialization technique to kernels other than those we have examined.

5. Adaptive Parallelism: GP-AUCB

While the analysis of the GP-BUCB algorithm in Sections 4.4 and 4.5 used feedback mappings $\text{fb}[t]$ specified by the problem instance, it may be useful to let the algorithm control when to request feedback, and to allow this feedback period to vary in some range not easily described by any constant B. For example, allowing the algorithm to control parallelism is desirable in situations where the cost of executing the algorithm's requested actions depends on both the number of batches and the number of individual actions or experiments in those batches. Consider a chemical experiment, in which the cost may depend on the time to complete the batch of reactions and the cost of the reagents needed for each individual experiment. In such a case, confronting an initial state of relative ignorance about the reward function, it may be desirable to avoid using a wasteful level of parallelism. Motivated by this, we develop an alternative to our requirement in GP-BUCB that $t - \text{fb}[t] \leq B$; we will instead specify a $C > 0$ and choose the feedback mapping $\text{fb}[t]$ in concert with the sequence of actions selected by the algorithm such that $I(f(\boldsymbol{x}); \boldsymbol{y}_{\text{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\text{fb}[t]}) \leq C, \forall \boldsymbol{x} \in D, \forall t \geq 1$. This requirement on $\text{fb}[t]$ in terms of C may appear stringent, but in fact it can be easily

satisfied by on-line, data-driven construction. The GP-AUCB algorithm adaptively controls feedback through precisely such a mechanism.

Section 5.1 introduces GP-AUCB and states a corollary regret bound for this algorithm. A few comments on local versus global stopping criteria for adaptivity of algorithms follow in Section 5.2.

5.1 GP-AUCB Algorithm

The key innovation of the GP-AUCB algorithm is in choosing $\text{fb}[t]$ online, using a limit on the amount of information hallucinated within the batch. Such adaptive batch length control is possible because we can measure online the amount of information hallucinated with respect to f using Equation (3), even in the absence of the observations themselves. This quantity can be used in a stopping condition; when it exceeds a pre-defined constant C, the algorithm terminates the batch and waits for the environment to return observations for the pending actions. The feedback mapping fb is then updated to include these new observations and the selection of a new batch begins. The resulting algorithm, GP-AUCB, is shown in Algorithm 4.

GP-AUCB is also applicable in the delay setting. In Section 3.1, a view of the delay setting was presented in which an algorithm maintains a queue of pending observations, where this queue is of size B and the algorithm submits a query in any round during which the queue is not full. This is natural for GP-BUCB, particularly if the delay on any observation is known to be bounded by B', i.e., $t - \text{fb}[t] \leq B'$; in such a case, choosing $B = B'$ gives an algorithm which submits an action every round. However, if B' is unknown, the queue size B would have to be chosen in some other way, such that potentially $B < B'$. In this case, the algorithm might have B pending observations at the beginning of a round, a full queue, and so decline to submit an action in that round, i.e., balk. Analogously, GP-AUCB in the delay setting implements a queue which is bounded by the conditional mutual information of the corresponding observations and f, given the current posterior. At each round, GP-AUCB checks if this quantity is more than a pre-defined value C, and only submits a query if it is not. Consequently, if $C < \gamma_{B'-1}$, the algorithm may balk on some rounds.

By terminating batches (or balking) such that no action is selected when the conditional information of the pending observations with respect to f is more than C, the GP-AUCB algorithm ensures that

$$I(f(\boldsymbol{x}); \boldsymbol{y}_{\text{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\text{fb}[t]}) \leq I(f; \boldsymbol{y}_{\text{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\text{fb}[t]}) \leq C, \ \forall \boldsymbol{x} \in D, \ \forall t \geq 1,$$

where t indexes all actions selected by the algorithm, the first inequality follows from the monotonicity of conditional mutual information, and the second inequality follows from the stopping condition. This result implies that Inequality (10) is satisfied, a key requirement of Theorem 2. In contrast, GP-BUCB satisfies the requirement that the second inequality hold by selecting a value for C greater than the conditional information which could be gained in *any* batch of a fixed size, as in Inequality (14), potentially resulting a choice of C larger than necessary for a given B. Since GP-AUCB considers the batches which are actually constructed, it can be expected to enable a higher level of parallelism for the same C, or a comparable level of parallelism for a smaller C.

It is also important to contrast the behavior of GP-AUCB with a scheduled, monotonically increasing level of parallelism. Under the stopping condition, the batch length is

Algorithm 4 GP-AUCB

Input: Decision set D, GP prior μ_0, σ_0, kernel $k(\cdot,\cdot)$, information gain threshold C.
Set $\text{fb}[t'] = 0,\ \forall t' \geq 1$, $G = 0$.
for $t = 1, 2, \ldots, T$ **do**
  if $G > C$ **then**
    Obtain $y_{t'} = f(\boldsymbol{x}_{t'}) + \varepsilon_{t'}$ for $t' \in \{\text{fb}[t-1], \ldots, t-1\}$
    Perform Bayesian inference to obtain $\mu_{t-1}(\cdot)$ via Equation (1)
    Set $G = 0$
    Set $\text{fb}[t'] = t-1,\ \forall t' \geq t$
  end if
  Choose $\boldsymbol{x}_t = \text{argmax}_{\boldsymbol{x} \in D}[\mu_{\text{fb}[t]}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})]$
  Set $G = G + \frac{1}{2}\log\left(1 + \sigma_n^{-2}\sigma_{t-1}^2(\boldsymbol{x}_t)\right)$
  Compute $\sigma_t(\cdot)$ via Equation (2)
end for

chosen in response to the algorithm's need to explore or exploit as dictated by the decision rule, Equation (7). This does tend to cause an increase in parallelism; the batch length may possibly become quite large as the shape of f is better and better understood and the variance of $f(\boldsymbol{x}_t)$ tends to decrease. However, if exploratory actions are chosen, the high information gain of these actions contributes to a relatively early arrival at the information gain threshold C and thus relatively short batch length, even late in the algorithm's run.

Since all actions are selected when $I(f; \boldsymbol{y}_{\text{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\text{fb}[t]}) \leq C$ for all $\boldsymbol{x} \in D$, this approach meets the conditions of Theorem 2, yielding the following corollary:

Corollary 6 *Let the GP-AUCB algorithm be employed with a specified constant $\delta \in (0,1)$ and a specified constant $C > 0$, for which the resulting feedback mapping fb : $\mathbb{N} \to \mathbb{N}$ guarantees $I(f; \boldsymbol{y}_{fb[t]+1:t-1} \mid \boldsymbol{y}_{1:fb[t]}) \leq C, \forall t \geq 1$. If the conditions of one case of Theorem 2 are met, and $\beta_t = \exp(2C)\alpha_{fb[t]+1}$ (Case 1 & 3) or $\beta_t = \exp(2C)\alpha_t$ (Case 2), then*

$$\Pr\left\{R_T \leq \sqrt{C_1 T \exp(2C)\alpha_T \gamma_T} + 2,\ \forall T \geq 1\right\} \geq 1 - \delta$$

where $C_1 = 8/\log(1 + \sigma_n^{-2})$.

Importantly, the specification of C directly specifies the regret bound under Corollary 6. Describing a problem in terms of C is thus natural in the case that we wish to parallelize an experimental process and our specification is what factor additional regret is acceptable, as compared to the sequential GP-UCB algorithm. The batch sizes or balking which result can then be regarded as those which follow from this specification.

Despite the advantages of this approach, C is abstract and less natural for an experimentalist to specify than a maximum batch size or delay length. However, some intuition with regard to C may be obtained. First, C can be selected to deliver batches with a specified minimum size B_{min}. To ensure this occurs, C can be set such that $C > \gamma_{(B_{min}-1)}$, i.e., no set of queries of size less than B_{min} could possibly gain enough information to end the batch. A satisfactory C can be found by either obtaining $\gamma_{(B_{min}-1)}$ directly (tractable for small B_{min}) or via a constant factor bound (Krause and Guestrin, 2005) using the amount

of information which could be gained during uncertainty sampling (Algorithm 3). Note that it is also possible to combine the results of Section 4.5 with Corollary 6 to produce a two-stage adaptive algorithm which can deliver high starting parallelism, very high parallelism as the run proceeds, and a low information gain bound C, yielding a favorable asymptotic regret bound. This may be done by initializing thoroughly enough that, for a pre-specified C and B_{min}, $\gamma^{\text{init}}_{B_{min}-1} < C$, such that the stopping condition cannot take effect until the batch size is at least B_{min}, and then running the GP-AUCB algorithm. This procedure ensures that all batches are of size at least B_{min} and no action is selected using more than C hallucinated information. Alternatively, for uninitialized GP-AUCB, note that C could be quite small, e.g., γ_1; a very small choice for C should produce GP-UCB-like, fully-sequential behavior while the algorithm knows very little, but as the algorithm begins repeatedly selecting actions within a small, well-characterized set, it will permit a greater level of parallelism.

In Section 3.1, the pessimistic and optimistic views of parallelism discussed therein could respectively be viewed as emphasizing one or the other of action selection or feedback receipt as the most important clock by which the system's progress could be judged. However, in the simple batch and delay settings, these perspectives were fixed to one another by the constant B, governing the maximum level of parallelism. Allowing adaptive or stochastic delay and balking breaks this fixed linkage and can be thought of as creating a third clock timing the *opportunities* for the algorithm to select a single action. If the delay is fixed in terms of the number of such opportunities between action and observation, rather than the number of actions between these events, this gives a more natural notion of waiting for observations and allows a better comparison of the tradeoffs inherent in such policies. In our experiments, this opportunity-for-action perspective is explicitly used for all adaptive algorithms shown in Figures 3 and 6, which apply the adaptive algorithms to the delay setting. We also take our previous, pessimistic or action-centered perspective in Figure 5 when looking at adaptive batch size selection, allowing examination of how much regret adaptive batch size selection incurs as compared to fully sequential of fixed parallel algorithms.

5.2 Locally Stopped Adaptive Algorithms

Recently, Azimi et al. (2012b) proposed the Hybrid Batch Bayesian Optimization algorithm (HBBO). HBBO implements a check on the faithfulness of a hallucinated posterior, similar to our approach. This check is expressed not in terms of information gain, but rather expected prediction error versus the true posterior if all information had been acquired. Their stopping condition is also only locally checked at the selected $\boldsymbol{x}_t$, rather than all $\boldsymbol{x}$ in D. Azimi et al. (2012b) employ this stopping condition along with a constraint that the size of the batch assembled can never exceed a pre-specified B_{max}. They show that, in practice, much of the time the algorithm is "safe" under the local faithfulness condition and the level of parallelism is actually controlled by B_{max}. In this section, we consider how our results similarly extend to local stopping conditions.

Theorem 2's requirement on the hallucinated conditional information gain is stated in terms of Equation (10), a bound on hallucinated information with respect to $f(\boldsymbol{x})$ for all $\boldsymbol{x} \in D$. Through Equation (9), this bound ensures that the confidence intervals used to select actions are still sufficiently faithful to those based on the true posterior, i.e., that

Algorithm 5 GP-AUCB Local

Input: Decision set D, GP prior μ_0, σ_0, kernel $k(\cdot,\cdot)$, information gain threshold C, maximum batch size B_{max}.
Set $\text{fb}[t'] = 0,\ \forall t' \geq 1$.
for $t = 1, 2, \ldots, T$ **do**
  if $t - \text{fb}[t] > B_{max}$ **or** $\exists\, \boldsymbol{x} \in D : \sigma_{\text{fb}[t]}(\boldsymbol{x})/\sigma_{t-1}(\boldsymbol{x}) > \exp(C)$ **then**
    Obtain $y_{t'} = f(\boldsymbol{x}_{t'}) + \varepsilon_{t'}$ for $t' \in \{\text{fb}[t-1], \ldots, t-1\}$
    Perform Bayesian inference to obtain $\mu_{t-1}(\cdot)$ via Equation (1)
    Set $\text{fb}[t'] = t-1,\ \forall t' \geq t$
  end if
  Choose $\boldsymbol{x}_t = \text{argmax}_{\boldsymbol{x} \in D}[\mu_{\text{fb}[t]}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})]$
  Compute $\sigma_t(\cdot)$ via Equation (2)
end for

$\sigma_{t-1}(\boldsymbol{x})$ does not become too small with respect to $\sigma_{\text{fb}[t]}(\boldsymbol{x})$. In the previous analysis, we ensured that this bound held for all $\boldsymbol{x} \in D$ by bounding $I(f; \boldsymbol{y}_{\text{fb}[t]+1:t-1} | \boldsymbol{y}_{1:\text{fb}[t]})$, an upper bound on each of the local information gains. However, in order to select actions, $\sigma_{t-1}(\boldsymbol{x})$ is calculated on-line; if D is of finite size, it is thus possible (if expensive) to compute the ratio $\sigma_{\text{fb}[t]}(\boldsymbol{x})/\sigma_{t-1}(\boldsymbol{x})$ for every $\boldsymbol{x}$ in D and every time step. Similar to GP-AUCB, it is possible to create an algorithm which uses Equation (7) to select actions and which terminates batches adaptively whenever there is any $\boldsymbol{x} \in D$ where this ratio is greater than $\exp(C)$ for a specified $C > 0$. Such an algorithm retains the regret bounds of Theorem 2. With the additional constraint that the assembled batch size not exceed a specified B_{max}, we denote this algorithm GP-AUCB Local and present it as Algorithm 5. We also test this algorithm in some of the experiments and figures in Section 7, along with HBBO.

A number of statements may be made regarding GP-AUCB Local. First, in the case of a flat prior, e.g., $f \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$, Equation (7) reduces to $x_t = \text{argmax}_{\boldsymbol{x} \in D}\, \sigma_{t-1}(\boldsymbol{x})$ until feedback is obtained at the end of the first batch, i.e., uncertainty sampling (Algorithm 3). GP-AUCB Local's first batch may thus contain a very large number of actions, broadly initializing the decision set. Such a procedure resembles the typical initialization of bandit algorithms and may be attractive in some settings, particularly those in which parallelism is essentially unlimited and the central concern is the number of batches. Second, in practice, nearly all of batches of GP-AUCB Local are stopped via the maximum batch size constraint because the largest local information gain may be small, even for a large batch. This means that this algorithm is effectively implementing GP-BUCB in the simple parallel case, where $B = B_{max}$, albeit with a tighter regret bound, since the specified C only needs to exceed the *local* information gain, rather than the maximum global information gain.

6. Lazy Variance Calculations

In this section, we introduce the notion of lazy variance calculations, which may be used to greatly accelerate the computation of many UCB-based algorithms, including GP-UCB, GP-BUCB, and GP-AUCB, without any loss of performance.

While the probabilistic inference carried out by GP-UCB, GP-BUCB, and GP-AUCB may be implemented in closed form, without the need for expensive approximate inference, the computational cost of the algorithms may still be high, particularly as the number of observations increases. In applications where a finite decision set is considered at every time t, the major computational bottleneck is calculating the posterior mean $\mu_{\mathrm{fb}[t]}(\boldsymbol{x})$ and variance $\sigma^2_{t-1}(\boldsymbol{x})$ for the candidate actions, as required to calculate the decision rule and choose an action $\boldsymbol{x}_t$. The mean is updated only whenever feedback is obtained, and—upon computation of the Cholesky factorization of $\mathbf{K}(X_{\mathrm{fb}[t]}, X_{\mathrm{fb}[t]}) + \sigma_n^2 I$—the calculation of the posterior mean $\mu_{\mathrm{fb}[t]}(\boldsymbol{x})$ takes $O(t)$ additions and multiplications. On the other hand, $\sigma^2_{t-1}(\boldsymbol{x})$ must be recomputed for every $\boldsymbol{x} \in D$ after every round, and requires solving backsubstitution, which requires $O(t^2)$ computations. For large decision sets D, the variance computation thus dominates the computational cost of GP-BUCB.

Fortunately, for any fixed decision $\boldsymbol{x}$, $\sigma^2_t(\boldsymbol{x})$ is non-increasing in t. This fact can be exploited to dramatically improve the running time of GP-BUCB. The key idea is that instead of recomputing $\sigma_{t-1}(\boldsymbol{x})$ for all candidate actions $\boldsymbol{x}$ in every round t, we can maintain an upper bound $\widehat{\sigma}_{t-1}(\boldsymbol{x})$, initialized to $\widehat{\sigma}_0(\boldsymbol{x}) = \infty$. In every round, we lazily apply the GP-BUCB rule with this upper bound to identify

$$\boldsymbol{x}_t = \operatorname*{argmax}_{\boldsymbol{x} \in D} \left[\mu_{\mathrm{fb}[t]}(\boldsymbol{x}) + \beta_t^{1/2} \widehat{\sigma}_{t-1}(\boldsymbol{x}) \right]. \tag{17}$$

We then recompute $\widehat{\sigma}_{t-1}(\boldsymbol{x}_t) \leftarrow \sigma_{t-1}(\boldsymbol{x}_t)$. If $\boldsymbol{x}_t$ still lies in the argmax of Equation (17), we have identified the next action to take, and set $\widehat{\sigma}_t(\boldsymbol{x}) = \widehat{\sigma}_{t-1}(\boldsymbol{x})$ for all $\boldsymbol{x} \in D$. Minoux (1978) proposed a similar technique, concerning calculating the greedy action for submodular maximization, which the above technique generalizes to the bandit setting. A similar idea was also employed by Krause et al. (2008) in the Gaussian process setting for experimental design. The lazy variance calculation method leads to dramatically improved empirical computational speed, discussed in Section 7.4. Note also that the quantities needed for a rank-1 update of the Cholesky decomposition of the observation covariance $\mathbf{K}(X_t, X_t) + \sigma_n^2 I$ are obtained at no additional cost; in order to select $\boldsymbol{x}_t$, we calculate the posterior standard deviation $\sigma_{t-1}(\boldsymbol{x}_t)$, which requires precisely these values.

Locally stopped algorithms (Section 5.2) may have stopping conditions which require $\sigma_{t-1}(\boldsymbol{x})$ for every $\boldsymbol{x} \in D$, which would seem to indicate that the lazy approach is not applicable. However, they may also benefit from lazy variance calculations. Since the global conditional information gain bounds the local information gain for all $\boldsymbol{x} \in D$, as in Inequality (11), we obtain the implication

$$I(f; \boldsymbol{y}_{\mathrm{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\mathrm{fb}[t]}) \leq C \implies \nexists\, \boldsymbol{x} \in D : I(f(\boldsymbol{x}); \boldsymbol{y}_{\mathrm{fb}[t]+1:t-1} \mid \boldsymbol{y}_{1:\mathrm{fb}[t]}) > C$$

that is, that until the stopping condition for GP-AUCB is met, the stopping condition for GP-AUCB Local is also not met, and thus no local calculations need be made. In implementing GP-AUCB Local, we may run what is effectively lazy GP-AUCB until the global stopping condition is met, at which time we transition to GP-AUCB Local. For a fixed maximum batch size B_{max}, it is often the case that local variance calculations become only very rarely necessary after the first few batches.

We have so far in this section concentrated on the case where D is of finite size. It is in general challenging to optimize the decision rule (a possibly multimodal function) over D if

D is a continuous set, as in Case 2 of Theorem 2. Many heuristics are reasonable, but any heuristic which re-uses candidate actions from round to round (e.g., one which considers repeating past actions $\boldsymbol{x}_{t'}, \forall t' < t$, or employs an expanding, finite discretization of D) could also be accelerated by this method.

7. Experiments

We compare GP-BUCB with several alternatives: (1) The strictly sequential GP-UCB algorithm ($B = 1$), which *immediately* receives feedback from each action without batching or delay, thus providing the baseline comparison from the pessimistic perspective (see Section 3.1); (2) Two versions of a state-of-the-art algorithm for Batch Bayesian optimization proposed by Azimi et al. (2010), which can use either a UCB or Maximum Expected Improvement (MEI) decision rule, herein SM-UCB and SM-MEI respectively. Note that the algorithm of Azimi et al. (2010) is not applicable to the delay setting and so does not appear in our delay experiments. Similarly, we compare GP-AUCB against two other adaptive algorithms: (1) HBBO, proposed by Azimi et al. (2012b), which checks an expected prediction error stopping condition, makes decisions using either an MEI or a UCB decision rule, and is applicable only to the batch setting; and (2) GP-AUCB Local, a local information gain-checking adaptive algorithm described in Section 5.2. We also present some experimental comparisons across these two sets of algorithms.

In Section 7.1, we describe the computational experiments in more detail. We perform each of these experiments for several data sets. These data sets and the corresponding experimental results are presented in Section 7.2. We highlight the optimistic perspective on parallelism and the tradeoffs inherent in adaptive parallelism in Section 7.3. Finally, we present the results of the computational time comparisons in Section 7.4.

7.1 Experimental Comparisons

We perform a number of different experiments using this set of algorithms: (1) A simple experiment in the batch case, in which the non-adaptive batch length algorithms are compared against one another, using a single batch length of $B = 5$ (Figure 2); (2) A corresponding experiment in the delay case using a delay of $B = 5$ rounds between action and the corresponding observation, comparing GP-UCB, GP-BUCB, GP-AUCB, and GP-AUCB Local against one another, where the two adaptive algorithms may balk (Figure 3); (3) An experiment examining how changes in the batch length over the range $B = 5, 10$, and 20 affect performance of the non-adaptive algorithms (Figure 4), and a similar experiment where the adaptive algorithms may terminate batches freely, with the restriction that batches must contain at least one and at most 5, 10, or 20 actions (Figure 5); (4) A corresponding experiment in the delay setting, examining how fixed delay length values of $5, 10$, and 20 rounds affect algorithm performance, and in which the adaptive algorithms may balk (Figure 6); (5) An experiment which examines how parallelism and different parameterizations of execution cost may be traded off (Figure 7); and (6) an experiment comparing execution time for various algorithms in the batch case, comparing basic and lazy versions (see Section 6) of the algorithms presented (Figure 8). In the interest of space, some plots are reserved to Online Appendix 2. We also present the results of the experiments in tabular form in Online Appendix 3. The algorithms do not receive an initialization set of observations in

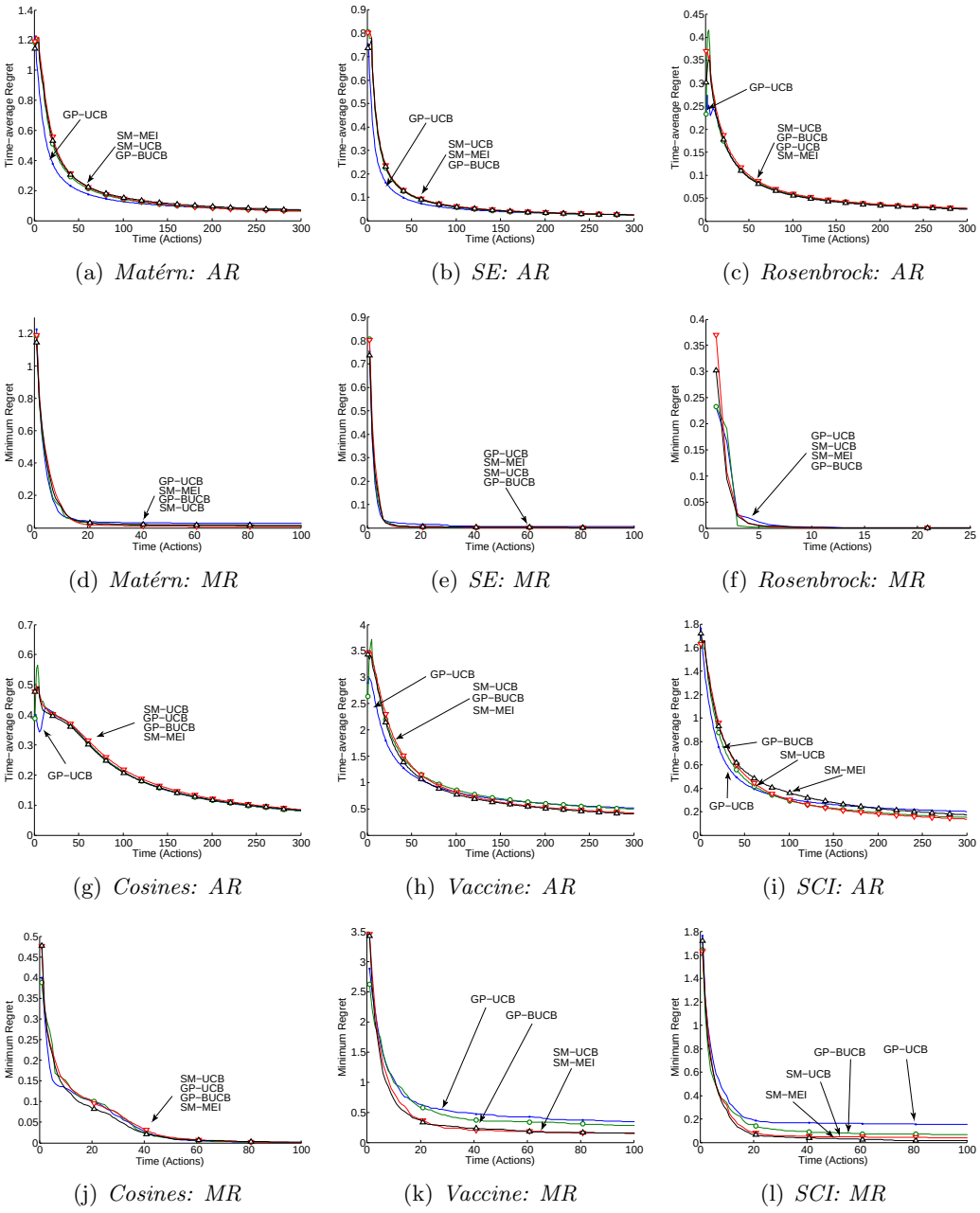
any of the experiments. All experiments were performed in MATLAB using custom code, which we make publicly available.[4]

Comparisons of reward and regret among the algorithms discussed above are presented in terms of their cumulative regret, as well as their simple regret (the function's maximum value minus the best reward obtained). Execution time comparisons are performed using wall-clock time elapsed since the beginning of the experiment, recorded at ends of algorithmic time steps. All experiments were repeated for 200 trials, with pseudo-independent tie-breaking and observation noise for each trial. Additionally, in those experimental cases where the reward function was a draw from a GP (the SE and Matérn problems), each trial used a pseudo-independent draw from the same GP.

In the theoretical analysis in Section 4, the crucial elements in proving the regret bounds of GP-BUCB and GP-AUCB are C, the bound on the information which can be hallucinated within a batch and β_t, the exploration-exploitation tradeoff parameter, which is set with reference to C to ensure confidence interval containment of the reward function. For practical purposes, it is often necessary to define β_t and the corresponding parameter of GP-UCB, α_t, in a fashion which makes the algorithm considerably more aggressive than the regret bound requires. This aggressiveness is particularly important in cases where each observation is very expensive. Setting α_t or β_t in this fashion removes the high-probability guarantees in the regret bound, but often produces excellent empirical performance. On the other hand, leaving the values for α_t and β_t as would be indicated by the theory results in heavily exploratory behavior and very little exploitation. In this paper, in all algorithms which use the UCB or BUCB decision rules, the value of α_t has been set such that it has a small premultiplier (0.05 or 0.1, see Table 2), yielding substantially smaller values for α_t. Further, despite the rigors of analysis explored above in Section 4, we choose to set $\beta_t = \alpha_{\mathrm{fb}[t]+1}$ for the batch and delay algorithms, without reference to the value of C or the batch length B. Taking either of these measures removes the guarantees of correctness as carefully crafted in Section 4. However, as verified by the experiments comparing batch sizes, this is often not a substantial detriment to performance, even for large batch sizes; the batch algorithms generally remain quite competitive with the sequential GP-UCB algorithm. This approach is additionally supported by interactions between local information gain and batch size constraints seen in practice with GP-AUCB Local. One experimental advantage of this approach is that (with some limitations necessitated by the adaptive algorithms) the various algorithms using a UCB decision rule are using the same exploration-exploitation tradeoff parameter at the same iteration, including GP-UCB, GP-BUCB, GP-AUCB, and even SM-UCB and HBBO when using the UCB decision rule. This choice enables us to remove a confounding factor in comparing how well the algorithms overcome the disadvantages inherent in the batch and delay settings.

In the adaptive algorithms (GP-AUCB and GP-AUCB Local), C still establishes the stopping condition, even though it is not used in setting β_t. For GP-AUCB, we specify a minimum batch size or acceptable number of queued observations B_{min} and use uncertainty sampling to calculate a constant-ratio upper bound on $\gamma_{B_{min}}$, as discussed in Section 4.4. Since the ratio $e/(e-1)$ in this bound is > 1, we also use a linear upper bound $\gamma_1 B_{min}$ and set C to the smaller of the two bounds. This choice ensures that the algorithm will always be

4. See `www.its.caltech.edu/~tadesaut/`.



(a) *Matérn: AR* (b) *SE: AR* (c) *Rosenbrock: AR*

(d) *Matérn: MR* (e) *SE: MR* (f) *Rosenbrock: MR*

(g) *Cosines: AR* (h) *Vaccine: AR* (i) *SCI: AR*

(j) *Cosines: MR* (k) *Vaccine: MR* (l) *SCI: MR*

Figure 2: Time-average (AR) and minimum (MR) regret, simple batch setting, batch size of 5. GP-UCB is shown in blue, GP-BUCB in green with circular markers, SM-MEI in black, with triangles, and SM-UCB red, with inverted triangles. When more than one algorithm name is associated with a single arrow, the vertical order of the labels indicates the local vertical order of the regret curves.

Problem Setting	Kernel Function	Hyperparameters	Noise Variance σ_n^2	Premultiplier (on α_t, β_t)
Matérn	covMaterniso	$l = 0.1$, $\sigma^2 = 0.5$	0.0250	0.1
SE	covSEiso	$l = 0.2$, $\sigma^2 = 0.5$	0.0250	0.1
Rosenbrock	RBF	$l^2 = 0.1$, $\sigma^2 = 1$	0.01	0.1
Cosines	RBF	$l^2 = 0.03$, $\sigma^2 = 1$	0.01	0.1
Vaccine	covLINone	$t_2 = 0.8974$	1.1534	0.05
SCI	covSEard	$l = [0.988, 1.5337, 1.0051, 1.5868]$, $\sigma^2 = 1.0384$	0.0463	0.1

Table 2: Experimental kernel functions and parameters.

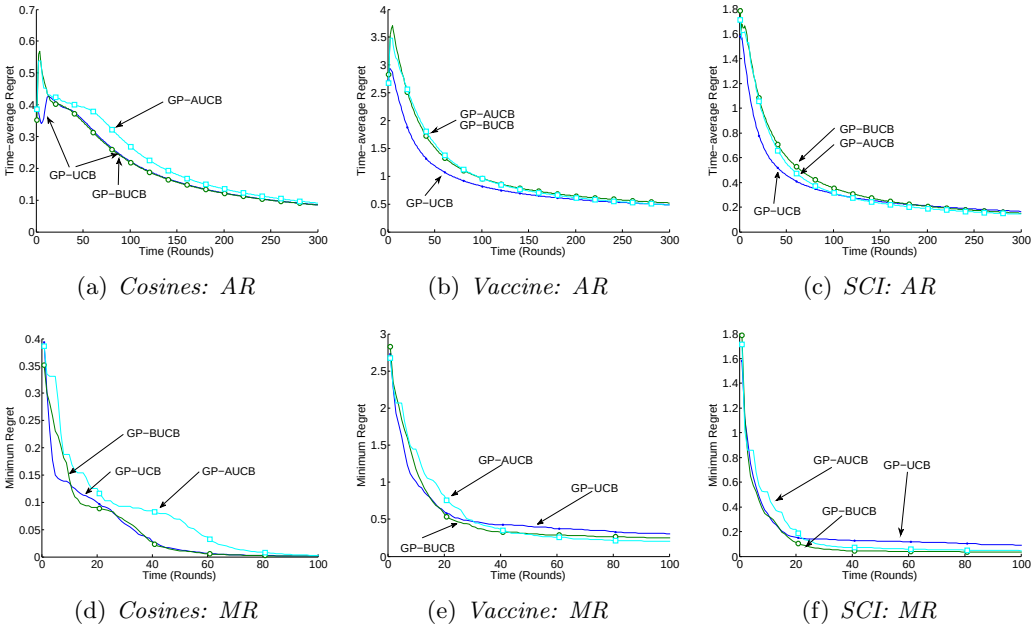
able to select at least B_{min} actions before receiving feedback. In GP-AUCB Local, it is more difficult to choose C appropriately, but we set $C = \max_{\boldsymbol{x} \in D} 1/2 \log(1 + B_{min}\sigma_n^{-2}\sigma_0^2(\boldsymbol{x}))$, where $\sigma_0^2(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x})$ is the prior variance at $\boldsymbol{x}$. This is the maximum information about any $f(\boldsymbol{x})$ which would result from noisily observing $f(\boldsymbol{x})$ B_{min} times. Since for both GP-AUCB and GP-AUCB Local we used B_{min} rather than $B_{min} - 1$ to set C, we implement the stopping condition using a strict inequality for the threshold, requiring that the information gain be $< C$ rather than $\leq C$. In experimental setting (3), we set $B_{min} = 1$, in line with HBBO, and in experimental settings (2), (4), and (5), we use $B_{min} = 2$.

7.2 Data Sets

We empirically evaluate GP-BUCB and GP-AUCB on several synthetic benchmark problems as well as two real applications. For each of the experimental data sets used in this paper, the kernel functions and experimental constants are listed in Table 2. Where applicable, the covariance function from the GPML toolbox (Ver. 3.1, Rasmussen and Nickisch, 2010) used is also listed by name. For all experiments, $\delta = 0.1$ (see Theorem 2) for UCB-based algorithms and tolerance $\epsilon = 0.02$ for HBBO. Each of the experiments discussed above is performed for each of the data sets described below and their results are presented, organized by experimental comparison (e.g., delay, adaptive batch size, etc.), in the accompanying figures.

7.2.1 Synthetic Benchmark Problems

We first test GP-BUCB and GP-AUCB in conditions where the true prior is known. A set of 100 example functions was drawn from a zero-mean GP with Matérn kernel over the interval $[0, 1]$. The kernel, its parameters, and the noise variance are known to each algorithm and D is the discretization of $[0, 1]$ into 1000 evenly spaced points. These experiments are also repeated with a Squared-Exponential kernel. Broadly speaking, these two problems are quite easy; the functions are fairly smooth, and for all algorithms considered, the optimum was found nearly every time, even for long batch sizes or delay lengths. For long batch lengths, substantial regret is incurred during the first batch, since no feedback is available; this is visible in Figures 11(a) and 11(b), in Online Appendix 2. For the batch lengths studied, the first batch of feedback provides a good localization of the optimum because the first few observations are highly informative; for this reason, subsequent values of the minimum regret are typically very small. For the same reason, average regret is largely driven by the length of the first batch. In the delay length experiments, the relative ease of the problems also means that the adaptive algorithms were able to use only relatively few actions and still obtain effective initialization.



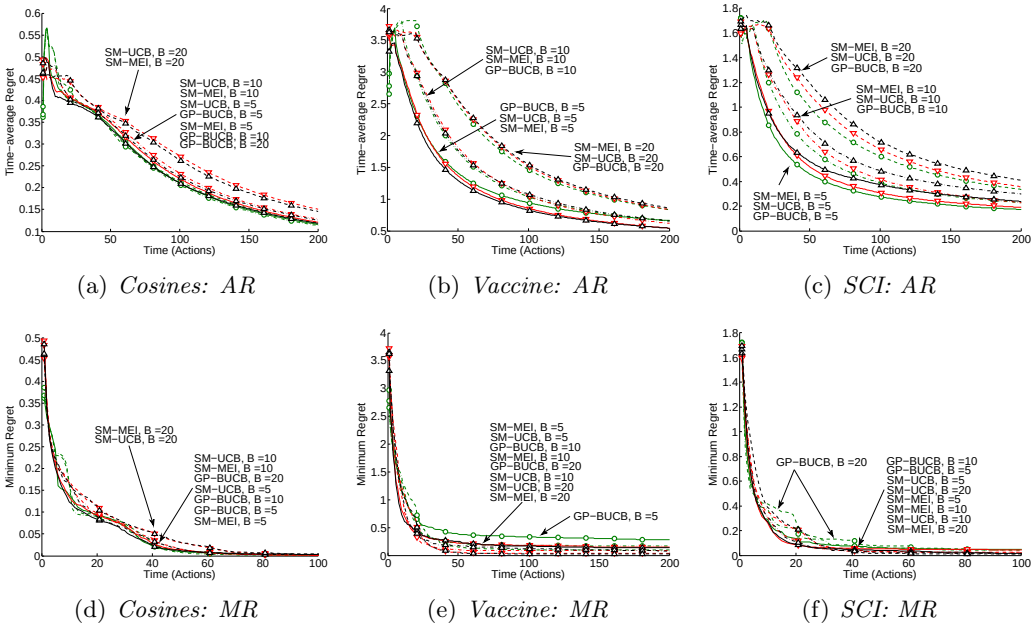
(a) *Cosines: AR* (b) *Vaccine: AR* (c) *SCI: AR*

(d) *Cosines: MR* (e) *Vaccine: MR* (f) *SCI: MR*

Figure 3: Time-average (AR) and minimum (MR) regret plots, delay setting, with a delay length of 5 rounds between action and observation. GP-AUCB is shown in cyan with square markers.

The Rosenbrock and Cosines test functions used by Azimi et al. (2010) are also considered, using the same Squared-Exponential kernel as employed in their experiments, though with somewhat different length scales. For both functions, D is a 31x31 grid of evenly-spaced points on $[0,1]^2$; D is thus similar in size to its counterpart in the Matérn and Squared-Exponential experiments. The values of the Rosenbrock test function at these points are heavily skewed toward the upper end of the reward range, such that the minimum regret is often nearly zero before the first feedback is obtained. In our experiments on the Rosenbrock function, similar performance was obtained across algorithms at each batch size in terms of both average and minimum regret. One result of interest is visible in Figure 6(c), which concerns delay length changes; it is possible to see that GP-AUCB balked too often in this setting, leading to substantial losses in performance relative to GP-AUCB Local and GP-BUCB. The Cosines test function also shows broadly similar results across specific problem instances, with only a small spread in regret among the algorithms tested. Because the Cosines function is multi-modal, the average regret seems to show two-phase convergence behavior, in which individual runs may be approaching local optima and subsequently finding the global optimum. The overly frequent balking by GP-AUCB present in the Rosenbrock test function is also present for longer delays in the Cosines function, as can be seen in 6(g).

In both delay experiments, this behavior may be explained by how the kernel chosen interacts with the stopping condition, which requires that the information gain with respect to the reward function f as a whole be less than a chosen constant C. With a flat prior,



(a) *Cosines: AR* (b) *Vaccine: AR* (c) *SCI: AR*

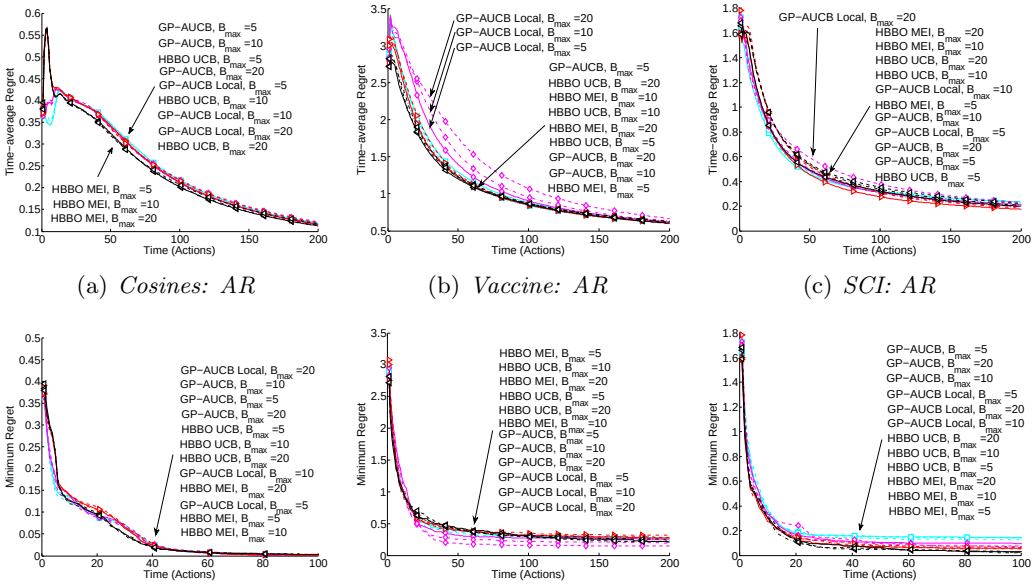
(d) *Cosines: MR* (e) *Vaccine: MR* (f) *SCI: MR*

Figure 4: Time-average (AR) and minimum (MR) regret plots, non-adaptive batch algorithms, batch sizes 5 (solid), 10 (dash-dot), and 20 (dashed).

GP-BUCB, GP-AUCB and GP-AUCB Local all initially behave like uncertainty sampling (see Sections 4.5 and 5.2). Since uncertainty sampling gains a great deal of information globally, GP-AUCB thus tends to balk; on the other hand, since uncertainty sampling scatters queries widely, the information gained with respect to any individual reward $f(\boldsymbol{x})$ may be comparatively small, and so GP-AUCB Local balks less or not at all. If the informativeness of the observations selected is overestimated, perhaps by poor specification of the long-range covariance properties of the assumed kernel function, this greater degree of balking by GP-AUCB may result in overall losses in performance.

7.2.2 Automated Vaccine Design

We also test GP-BUCB and GP-AUCB on a database of Widmer et al. (2010), as considered for experimental design by Krause and Ong (2011). This database describes the binding affinity of various peptides with a Major Histocompatibility Complex (MHC) Class I molecule, of importance when designing vaccines to exploit peptide binding properties. Algorithmic parallelization in such broad chemical screens is particularly attractive because automated, parallel equipment for carrying out these experiments is available. Each of the peptides which bound with the MHC molecule is described by a set of chemical features in $\mathbb{R}^{45}$, where each dimension corresponds to a chemical feature of the peptide. The binding affinity of each peptide, which is treated as the reward or payoff, is described as an offset IC_{50} value. The experiments use an isotropic linear kernel fitted on a different MHC molecule from the same data set. Since the data describe a phenomenon which has a mea-



(a) *Cosines: AR* (b) *Vaccine: AR* (c) *SCI: AR*

(d) *Cosines: MR* (e) *Vaccine: MR* (f) *SCI: MR*

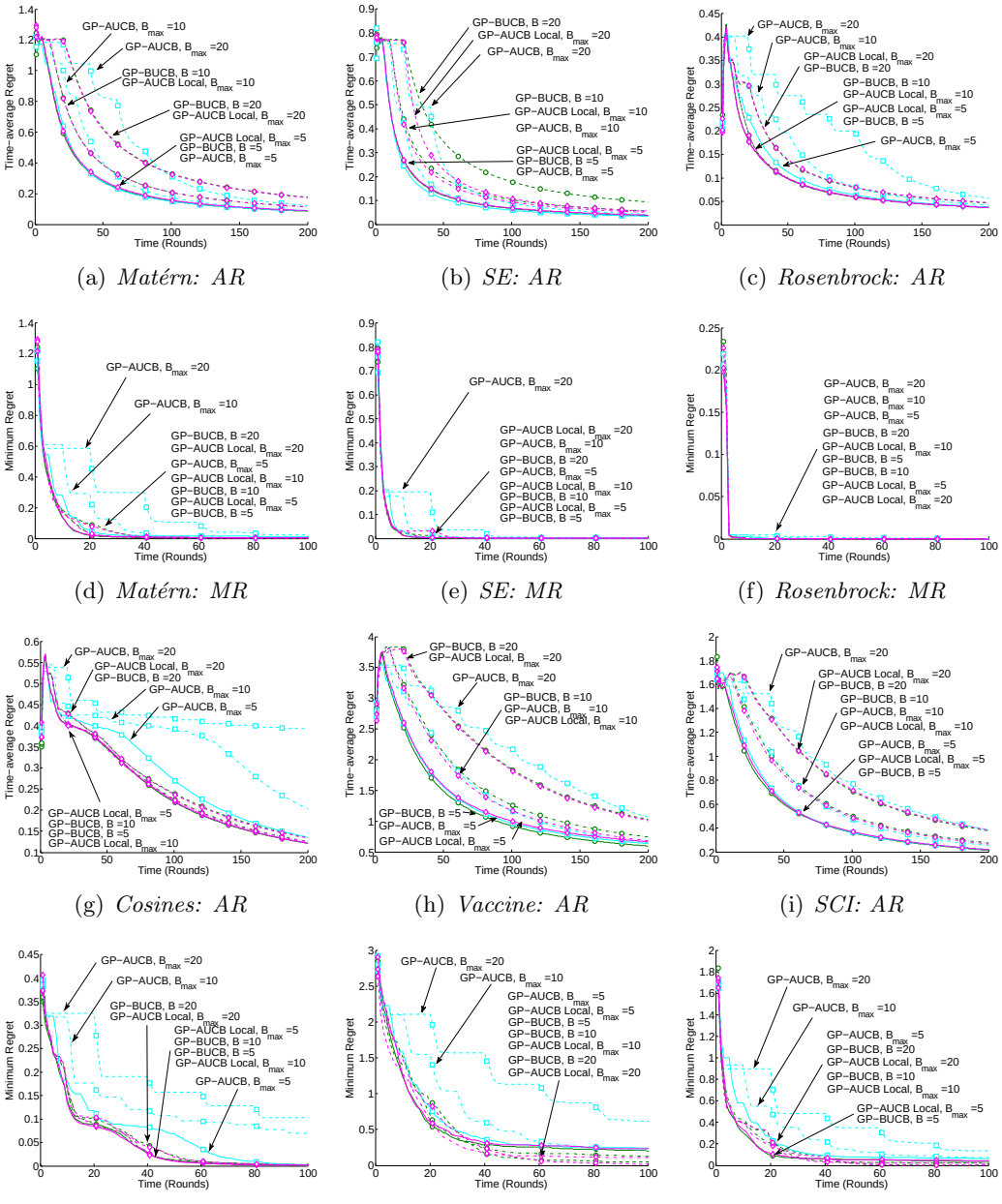
Figure 5: Time-average (AR) and minimum (MR) regret plots, adaptive batch algorithms, maximum batch sizes 5, 10, and 20. HBBO is shown in black with left pointing triangle makers when using an MEI decision rule and in red with right pointing triangle makers when using a UCB decision rule, while GP-AUCB Local is shown in pink with diamond markers. For the adaptive algorithms, minimum batch size B_{min} was set to 1, as in HBBO. The algorithms tended to run fully sequentially at the beginning, but quite rapidly switched to maximal parallelism.

surable limit, many members of the data set are optimal; out of 3089 elements of D, 124, or about 4%, are in the maximizing set. In the simple batch experiments, Figures 2(h) and 2(k), GP-BUCB performs competitively with SM-MEI and SM-UCB, both in terms of average and minimum regret, and converges to the performance of GP-UCB. In the simple delay setting, Figures 3(b) and 3(e), both GP-BUCB and GP-AUCB produce superior minimum regret curves to that of GP-UCB, while performing comparably in terms of long-run average regret; this indicates that the more thorough initialization of GP-AUCB and GP-BUCB versus GP-UCB may enable them to avoid early commitment to the wrong local optimum, thus finding a member of the maximizing set more consistently. This is consistent with the results of the non-adaptive batch size comparison experiment, Figures 4(b) and 4(e), which shows that as the batch size B grows, the algorithm must pay more "up front" due to its more enduring ignorance, but also tends to avoid missing the optimal set entirely. This same sort of tradeoff of average regret against minimum regret is clearly visible for the GP-AUCB Local variants in the experiments sweeping maximal batch size for adaptive algorithms, Figures 5(b) and 5(e).

7.2.3 Spinal Cord Injury (SCI) Therapy

Lastly, we compare the algorithms on a data set of leg muscle activity triggered by therapeutic spinal electrostimulation in spinal cord injured rats. From the 3-by-9 grid of electrodes on the array, a pair of electrodes is chosen to activate, with the first element of the pair used as the cathode and the second used as the anode. Electrode configurations are represented in $\mathbb{R}^4$ by the cathode and anode locations on the array. These active array electrodes create an electric field which may influence both incoming sensory information in dorsal root processes and the function of interneurons within the spinal cord, but the precise mechanism of action is poorly understood. Since the goal of this therapy is to improve the motor control functions of the lower spinal cord, the designated experimental objective is to choose the stimulus electrodes which maximize the resulting activity in lower limb muscles, as measured by electromyography (EMG). Batch or delay algorithms are particularly suited to this experimental setting because the time to process the EMG information needed to assess experimental stimuli may be quite long as compared to the time required to actually test a stimulus, and because idle time during the experimental session should be avoided to the degree possible. We use data with a stimulus amplitude of 5 V and seek to maximize the peak-to-peak amplitude of the recorded EMG waveforms from the right medial gastrocnemius muscle in a time window corresponding to a single interneuronal delay. This objective function attempts to measure the degree to which the selected stimulus activates interneurons controlling reflex activity in the spinal gray matter. This response signal is non-negative and for physical reasons does not generally rise above 3 mV. A Squared-Exponential ARD kernel was fitted using experimental data from 12 days post-injury. Algorithm testing is done using an reward function composed of data from 116 electrode pairs tested on the 14th day post-injury.

Like the Vaccine data set, the SCI data set displays a number of behaviors which indicate that the problem instance is difficult; in particular, the same tendency that algorithms which initialize more thoroughly eventually do better in both minimum and average regret was observed. This tendency is visible in the simple batch setting (Figures 2(i) and 2(l)), where GP-UCB is not clearly superior to either GP-BUCB or GP-AUCB. This is surprising because the pessimistic perspective on parallelism suggests that being required to work in batches, rather than one query at a time, might be expected to give the algorithm less information at any given round, and should thus be a disadvantage. This under-exploration in GP-UCB may be a result of the exploration-exploitation tradeoff parameter α_t being chosen to promote greater aggressiveness across all algorithms. Interestingly, this data set also displays both a small gap between the best and second-best values of the reward function (approximately 0.9% of the range) and a large gap between the best and third-best (approximately 7% of the range). When examining how many out of the individual experimental runs simulated selected $\boldsymbol{x}^* = \text{argmax}_{\boldsymbol{x} \in D} f(\boldsymbol{x})$ on the 200th query in the simple batch case, only 20% of GP-UCB runs choose $\boldsymbol{x}^*$; the numbers are considerably better for GP-BUCB, SM-UCB, and SM-MEI, at 35%, 30.5%, and 36%, but are still not particularly good. If the first suboptimal action is also included, these numbers improve substantially, to 63.5% for GP-UCB and 84%, 91%, and 96.5% for GP-BUCB, SM-UCB, and SM-MEI. These results indicate that the second-most optimal $\boldsymbol{x}$ is actually easier to find than the most optimal, to a fairly substantial degree. It is also important to place these results in the context of the



(a) *Matérn: AR* (b) *SE: AR* (c) *Rosenbrock: AR*

(d) *Matérn: MR* (e) *SE: MR* (f) *Rosenbrock: MR*

(g) *Cosines: AR* (h) *Vaccine: AR* (i) *SCI: AR*

(j) *Cosines: MR* (k) *Vaccine: MR* (l) *SCI: MR*

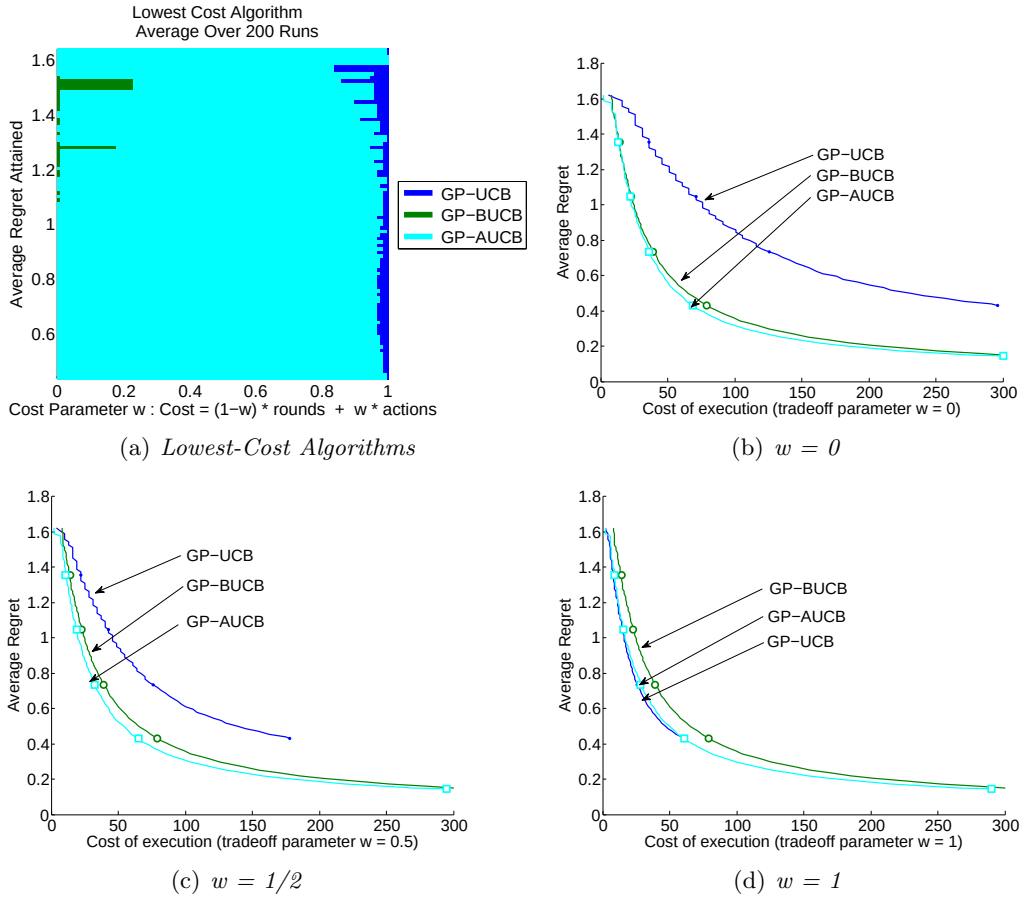
Figure 6: Time-average (AR) and minimum (MR) regret plots, delay setting, with delay lengths of 5, 10, and 20 rounds between action and observation. This experiment examines the degree to which these algorithms are able to cope with long delays between action and observation. Note that the adaptive algorithms, GP-AUCB and GP-AUCB Local, may balk at some rounds. The time-average regret is calculated with respect to the number of actions actually executed as of that round; this means that the number of queries submitted as of any particular round is hidden with respect to the plots shown, and may vary across runs of the same algorithm.

experimental setting; even assuming that the measured response values are reflective of a difference in spinal excitability between these two highest-performing stimuli, it may be that this very small difference in excitability would not yield any difference in therapeutic outcome. Since all of GP-BUCB, SM-UCB, and SM-MEI more consistently found one of the two best actions in the decision set than GP-UCB, all of them show strong performance in comparison to GP-UCB.

7.3 Parallelism: Costs and Tradeoffs

We have presented several algorithms, but an important question is which should be chosen to control any particular experimental process. Our motivation in pursuing parallel algorithms is the setting in which there is a cost—not accumulated in the regret—associated with the experimental process, such that each round or opportunity to submit a query is expensive, but the additional marginal cost of taking an action at that round is not very large. It is interesting to consider more precisely what we mean by "expensive" or "not very large," and also what effect varying these costs with respect to one another might have on which algorithm or level of parallelism is appropriate. In particular, one would expect a low level of parallelism to be beneficial if per-action costs are much higher than per-opportunity (i.e., when speed is less important than economy), while a high level of parallelism would be beneficial if the opposite is true, with intermediate levels of parallelism being superior in the middle. This intuition can be tested by measuring the costs and regret incurred by several algorithms solving the same problem. It is necessary to have a measure by which the performance of different algorithms can be compared, given a particular parameterization of costs. Here, we use an experiment in the delay setting, where the algorithm chooses to either take an action or balk at each round, and employ the average total cost up to the round in which a given average regret is first obtained.

Given N sample runs, a successful algorithm should have a (nearly) monotonically decreasing average regret curve, defined as $\bar{r}(T) = 1/N \sum_{n=1}^{N} R_{T,n}/T$, where $R_{T,n}$ is the cumulative regret of run n after T rounds; these regret curves are the same ones presented in previous experiments. After averaging over many runs, this curve can be inverted to find the first round $\tau(\bar{r})$ in which the sample average regret is at or below a particular $\bar{r}$. The average cost of running the algorithm until round $\tau(\bar{r})$ can then be computed. The cost of run n is the sum of two contributions, the first for running $\tau(\bar{r})$ rounds of the algorithm and the second for the actual execution of $a_n(\tau(\bar{r}))$ actions, where the number of actions executed varies depending on the data acquired. Parameterizing the relative costs of each round and each action using w, the average cost $C(\bar{r}, w) = (1 - w)\tau(\bar{r}) + w \cdot \bar{a}(\tau(\bar{r}))$ corresponding to a particular average regret value $\bar{r}$ can be obtained, where $\bar{a}(\tau(\bar{r})) = 1/N \sum_{n=1}^{N} a_n(\tau(\bar{r}))$. Note that $w \in [0, 1]$ translates to any constant, non-negative ratio of the cost of a single action to that of a single round. This procedure is not equivalent to fixing a value of $\bar{r}$, running each sample run of the algorithm until $R_{t,n}/t \leq \bar{r}$ and averaging over the costs incurred in so doing; in particular, if an algorithm has a non-zero probability of failing to ever obtain $\bar{r}$, individual sample runs may not terminate, making sensible comparison impossible. The calculation of $C(\bar{r}, w)$ as proposed here is robust to this case, giving an estimate of the expected cost to run the algorithm until a round in which the expected cumulative average regret is $\leq \bar{r}$.



(a) *Lowest-Cost Algorithms* (b) $w = 0$

(c) $w = 1/2$ (d) $w = 1$

Figure 7: Parameterized cost comparison on the SCI data set, simple delay case, $B = 5$. The same experiment is also presented in Figure 3(c), but in that figure, we take the pessimistic perspective and compare GP-BUCB and GP-AUCB with GP-UCB, where GP-UCB receives feedback every round. Here, we take the optimistic perspective, which treats parallelism as a potential advantage, and impose the same delay on all algorithms. Figure 7(a): the space of cost-tradeoff parameter w and attained average regrets $\bar{r}$ is colored according to which algorithm has the lowest mean cost at the round in which the mean, time-average regret is first $\leq \bar{r}$. Figures 7(b), (c), and (d) show $\bar{r}$ as a function of C and correspond to vertical slices through Figure 7(a) at the left, center, and right. Since GP-AUCB and GP-UCB pass on some rounds, the terminal cost of GP-AUCB and GP-UCB is possibly < 300.

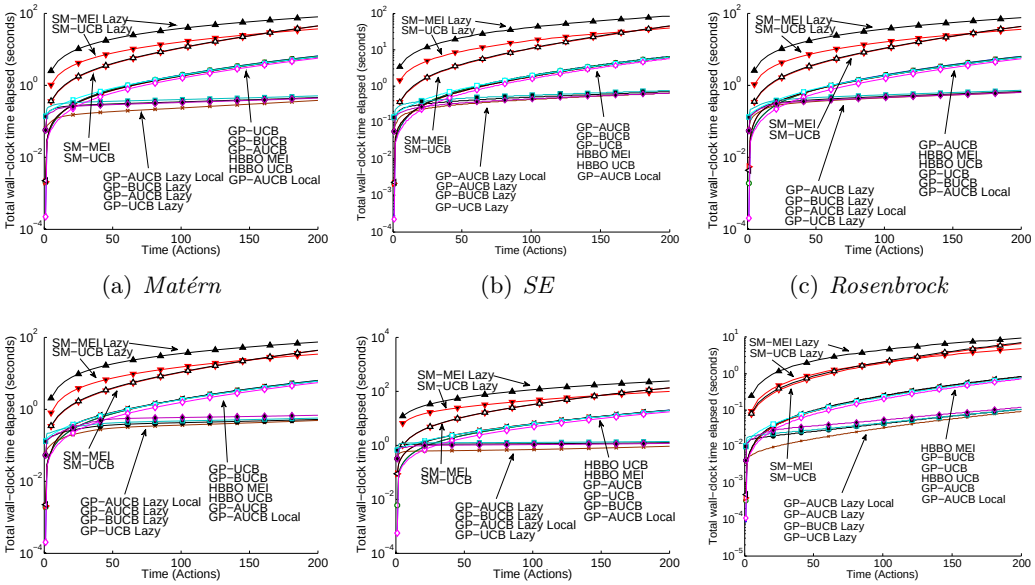
Among a set of algorithms, and given a test problem, one can find which among them has the lowest value of $C(\bar{r}, w)$ at any particular point in the $\bar{r}, w$ space. Similarly, for any fixed value of w, it is possible to once more invert the function and plot $\bar{r}_w(C)$; this plot resembles conventional average regret plots, and corresponds to intersections of each algorithm's $C(\bar{r}, w)$ surface with the plane at a fixed w.

We compare GP-BUCB, GP-AUCB, and GP-UCB in the SCI therapy setting, with a simple delay ($B = 5$). In this setting, GP-BUCB selects an action every round (filling its queue of pending experiments to 5, and then keeping it full) and GP-AUCB may balk, but will tend to fill its queue fully by the end of the experiment. Note that here, we employ GP-UCB under the same feedback mapping as the other algorithms, rather than its use as a benchmark in all of our previous experiments; it thus only submits an action when its queue of pending observations is empty, i.e., every fifth round. The results of this experiment are shown in Figure 7. In this scenario, GP-AUCB costs the least through most of the parameter space, due to its tendency to pass in early rounds, when the potential for exploitation is lowest. In line with the intuition described at the beginning of this section, the advantage changes to the fully sequential algorithm when w is large (i.e., parallelism is expensive), and to GP-BUCB when w is small. Many real-world situations lie somewhere between these extremes, suggesting that GP-AUCB may be useful in a variety of scenarios.

7.4 Computational Performance

We also examined the degree to which lazy variance calculations, as described in Section 6, reduce the computational overhead of each of the algorithms discussed. These results are presented in Figure 8. Note that for algorithms which appear as both lazy and non-lazy versions, the only functional difference between the two is the procedure by which the action is selected, not the action selection itself; all computational gains are without sacrificing accuracy and without requiring any algorithmic approximations. All computational time experiments were performed on a desktop computer (quad-core Intel i7, 2.8 GHz, 8 GB RAM, Ubuntu 10.04) running a single MATLAB R2012a process.

For all data sets, the algorithms lie in three broad classes: Class 1, comprised of the lazy GP-UCB family of algorithms; Class 2, the non-lazy versions of the GP-UCB family of algorithms, as well as the HBBO UCB and MEI variants; and Class 3, consisting of the SM-MEI and SM-UCB algorithms, in both lazy and non-lazy versions. Class 1 algorithms run to completion about one order of magnitude faster than those in Class 2, which in turn are about one order of magnitude faster than those in Class 3. The various versions of the simulation matching algorithm of Azimi et al. (2010) require multiple samples from the posterior over f to aggregate together into a batch, the composition of which is intended to match or cover the performance of the corresponding sequential UCB or MEI algorithm. The time difference between Class 2 and Class 3, approximately one order of magnitude, reflects the choice to run 10 such samples. Within Class 3, our implementation of the lazy version of SM-MEI is slower than the non-lazy version, largely due to the increased overhead of sorting the decision rule and computing single values of the variance; a more efficient implementation of either or both of these elements could perhaps improve on this tradeoff. The lazy algorithms also tend to expend a large amount of computational time early, computing upper bounds on later uncertainties, but tend to make up for this early investment later; this is even visible with regard to the lazy version of SM-UCB, which is initially slower than the non-lazy version, but scales better and, in all six data sets examined, ends up costing substantially less computational time by the 200th query.



(a) *Matérn* (b) *SE* (c) *Rosenbrock*

(d) *Cosines* (e) *Vaccine* (f) *SCI*

Figure 8: Elapsed computational time in batch experiments, B = 5. Lazy versions of algorithms (except GP-UCB) are shown will filled markers. Note the logarithmic vertical scaling in all plots. Note also the substantial separation between the three groups of algorithms, discussed in Section 7.4.

8. Conclusions

We develop the GP-BUCB and GP-AUCB algorithms for parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization. We present a unified theoretical analysis, hinging on a natural notion of conditional mutual information accumulated while making selections without observing feedback. Our analysis allows us to bound the regret of GP-BUCB and GP-AUCB, as well as similar GP-UCB-type algorithms. In particular, Theorem 2 provides high-probability bounds on the cumulative regret of algorithms in this class, applicable to both the batch and delay setting. These bounds also imply bounds on the convergence rate of such algorithms. Further, we prove Theorem 5, which establishes a regret bound for a variant GP-BUCB using uncertainty sampling as initialization. Crucially, this bound scales *independently* of the batch size or delay length B, if B is constant or polylogarithmic in T. Finally, we introduce lazy variance calculations, which dramatically accelerate computational performance of GP-based active learning decision rules.

Across the experimental settings examined, GP-BUCB and GP-AUCB performed comparably with state-of-the-art parallel and adaptive parallel Bayesian optimization algorithms, which are not equipped with theoretical bounds on regret. GP-BUCB and GP-AUCB also perform comparably to the sequential GP-UCB algorithm, indicating that GP-BUCB and GP-AUCB successfully overcome the disadvantages of only receiving delayed or batched feedback. As the family of algorithms we describe offers a spectrum of parallelism, we also

examine a parameterization of cost to achieve a given level of regret. In this comparison, GP-AUCB appears to offer substantial advantages over the fully parallel or fully sequential approaches. We believe that our results provide an important step towards solving complex, large-scale exploration-exploitation tradeoffs.

Acknowledgments

The authors thank Daniel Golovin for helpful discussions, the Edgerton laboratory at UCLA for the SCI data set, and the anonymous reviewers for their careful reading and many helpful suggestions. This work was partially supported by NIH project R01 NS062009, SNSF grant 200021_137971, NSF IIS-0953413, DARPA MSEE FA8650-11-1-7156, ERC StG 307036, a Microsoft Research Faculty Fellowship (AK) and a ThinkSwiss Research Scholarship (TD).

Appendix A. Proof of Theorem 2

In order to prove Theorem 2, this appendix first establishes a series of supporting lemmas. For clarity of development, we present the proof of the first case in detail, followed by the required lemmas and modifications required to prove the second and third cases. Since our three cases are those treated by Srinivas et al. (2012) for the GP-UCB algorithm, our proofs use Proposition 1 to generalize their theoretical analysis to the batch and delay cases. In the following, $\mu_{t-1}(\boldsymbol{x})$ and $\sigma_{t-1}(\boldsymbol{x})$ are found via Equations (1) and (2), which assume i.i.d. Gaussian noise of variance σ_n^2, even in Case 3, where the actual noise is non-Gaussian.

A.1 Case 1: Finite D

In all three cases, the first component of the proof is the establishment of confidence intervals which contain the payoff function f with high-probability. In Case 1, this is done by using a result established by Srinivas et al. (2012), presented here as Lemma 7.

Lemma 7 *(Lemma 5.1 of Srinivas et al., 2012) Specify $\delta \in (0,1)$ and set $\alpha_t = 2\log(|D|\pi_t/\delta)$, where $\sum_{t=1}^{\infty} \pi_t^{-1} = 1, \pi_t > 0$. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots \in D$ be an arbitrary sequence of actions. Then,*

$$P(|f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})| \leq \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x}),\, \forall \boldsymbol{x} \in D, \forall t \geq 1) \geq 1 - \delta.$$

Proof For $a \sim \mathcal{N}(0,1)$, $P(a > c) \leq 1/2\exp(-c^2/2)$. Conditioned on actions $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}\}$ and corresponding observations $\{y_1, \ldots, y_{t-1}\}$, $f(\boldsymbol{x}) \sim \mathcal{N}(\mu_{t-1}(\boldsymbol{x}), \sigma_{t-1}^2(\boldsymbol{x}))$; for any $\alpha_t > 0$,

$$P\left(\frac{f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})}{\sigma_{t-1}(\boldsymbol{x})} > \alpha_t^{1/2}\right) = P\left(\frac{f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})}{\sigma_{t-1}(\boldsymbol{x})} < -\alpha_t^{1/2}\right) \leq \frac{1}{2}\exp(-\alpha_t/2).$$

Note that these two events are the two ways the confidence interval on $f(\boldsymbol{x})$ could fail to hold, i.e., that $f(\boldsymbol{x}) \notin [\mu_{t-1}(\boldsymbol{x}) - \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x}),\, \mu_{t-1}(\boldsymbol{x}) + \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x})]$. Union bounding these confidence interval failure probabilities over D, $P(\exists \boldsymbol{x} \in D : f(\boldsymbol{x}) \notin [\mu_{t-1}(\boldsymbol{x}) - \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x}),\, \mu_{t-1}(\boldsymbol{x}) + \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x})]) \leq |D|\exp(-\alpha_t/2)$. Let $\delta/\pi_t = |D|\exp(-\alpha_t/2)$, implicitly defining α_t as specified. Union bounding in time and taking the complement yields

$$P(|f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})| \leq \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \in \{1, \ldots, T\}) \geq 1 - \delta\sum_{t=1}^{T} \pi_t^{-1}.$$

If $\pi_t > 0$ is chosen such that $\sum_{t=1}^{\infty} \pi_t^{-1} \leq 1$, the result follows. ■

This series convergence condition on π_t corresponds to a requirement that α_t grow sufficiently fast as to make confidence interval failures vanishingly unlikely as $t \to \infty$. Lemma 7 also implies that for $\mathcal{S}$, a subset of the positive integers, $P(|f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})| \leq \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \in \mathcal{S}) \geq 1 - \delta$, since $\pi_t > 0 \implies \sum_{t \in \mathcal{S}} \pi_t^{-1} \leq 1$.

Next, we must establish a link between confidence intervals which use a fully updated posterior and for which we have high probability guarantees of correctness (e.g., those in in Lemma 7), and the confidence intervals used in Equation (7), which use a hallucinated posterior. Lemma 8 shows that a bound on the local information hallucinated during the batch implies such a link between batch and sequential confidence intervals.

Lemma 8 *Suppose that at round t, there exists $C > 0$ such that*

$$I(f(\boldsymbol{x}); \boldsymbol{y}_{fb[t]+1:t-1} \mid \boldsymbol{y}_{1:fb[t]}) \leq C, \ \forall \boldsymbol{x} \in D. \tag{18}$$

Choose

$$\beta_t = \exp(2C)\alpha_{fb[t]+1}, \tag{19}$$

where Equation (6) relates sequential confidence intervals $C^{seq}_{fb[t]+1}(\boldsymbol{x})$ with the parameter $\alpha_{fb[t]+1}$ and Equation (8) relates batch confidence intervals $C^{batch}_t(\boldsymbol{x})$ with the parameter β_t. If $f(\boldsymbol{x}) \in C^{seq}_{fb[t]+1}(\boldsymbol{x})$, for all $\boldsymbol{x} \in D$, then $f(\boldsymbol{x}) \in C^{batch}_{t'}(\boldsymbol{x})$ for all $\boldsymbol{x} \in D$ and all t' such that $fb[t] + 1 \leq t' \leq t$.

Proof Noting that the confidence intervals $C^{\text{seq}}_{\text{fb}[t]+1}(\boldsymbol{x})$ and $C^{\text{batch}}_t(\boldsymbol{x})$ are both centered on $\mu_{\text{fb}[t]}(\boldsymbol{x})$,

$$C^{\text{seq}}_{\text{fb}[t]+1}(\boldsymbol{x}) \subseteq C^{\text{batch}}_t(\boldsymbol{x}), \ \forall \boldsymbol{x} \in D \iff \alpha^{1/2}_{\text{fb}[t]+1}\sigma_{\text{fb}[t]}(\boldsymbol{x}) \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \ \forall \boldsymbol{x} \in D.$$

By the definition of the conditional mutual information with respect to $f(\boldsymbol{x})$, and by employing Equation (18), Equation (9) follows. Choosing β_t as in Equation (19), it follows that

$$\alpha^{1/2}_{\text{fb}[t]+1}\sigma_{\text{fb}[t]}(\boldsymbol{x}) = \beta_t^{1/2} \exp(-C) \cdot \sigma_{\text{fb}[t]}(\boldsymbol{x}) \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}),$$

where the inequality is based on Equation (9), thus implying $C^{\text{seq}}_{\text{fb}[t]+1}(\boldsymbol{x}) \subseteq C^{\text{batch}}_t(\boldsymbol{x}) \forall \boldsymbol{x} \in D$. In turn, if $f(\boldsymbol{x}) \in C^{\text{seq}}_{\text{fb}[t]+1}(\boldsymbol{x})$, then $f(\boldsymbol{x}) \in C^{\text{batch}}_t(\boldsymbol{x})$. Further, since Equation (19) relates β_t to $\alpha_{\text{fb}[t]+1}$, then $\beta_{t'} = \beta_t$ for all $t' \in \{\text{fb}[t] + 1, \ldots, t\}$. Since $\sigma_{t'}(\boldsymbol{x})$ is non-increasing, $C^{\text{batch}}_{t'}(\boldsymbol{x}) \supseteq C^{\text{batch}}_t(\boldsymbol{x})$ for all such t', completing the proof. ■

With a bound C on the conditional mutual information gain with respect to $f(\boldsymbol{x})$ for any $\boldsymbol{x} \in D$, as in Equation (18), Lemma 8 links the confidence intervals and GP-BUCB decision rule at time t with the GP posterior after observation fb$[t]$. Lemma 9 extends this link to all $t \geq 1$ and all $\boldsymbol{x} \in D$, given a high-probability guarantee of confidence interval correctness at the beginning of all batches.

Lemma 9 *Let there exist a constant $C > 0$, a sequence of actions $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}\}$, and a feedback mapping $fb[t]$ such that for all $\boldsymbol{x} \in D$*

$$I(f(\boldsymbol{x}); \boldsymbol{y}_{fb[t]+1:t-1} \mid \boldsymbol{y}_{1:fb[t]}) \leq C, \ \forall t \geq 1.$$

Let $\beta_t = \exp(2C)\alpha_{fb[t]+1}$, $\forall t \geq 1$; then

$$P(f(\boldsymbol{x}) \in C^{seq}_{fb[t]+1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1) \geq 1 - \delta$$
$$\implies P(f(\boldsymbol{x}) \in C^{batch}_{t}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1) \geq 1 - \delta.$$

Proof For every $t \geq 1$, there exists a $\tau \geq 0$ such that $\tau = \text{fb}[t]$; let $\mathcal{S} = \{\tau_1, \tau_2, \ldots\}$ be the set of all such images under fb, such that $\text{fb}[t] \in \mathcal{S}$ for all $t \geq 1$. If β_t is chosen as specified, then for any t and $\tau = \text{fb}[t]$, if $f(\boldsymbol{x}) \in C^{\text{seq}}_{\tau+1}(\boldsymbol{x})$, Lemma 8 implies that $f(x) \in C^{\text{batch}}_t(\boldsymbol{x})$. If $f(\boldsymbol{x}) \in C^{\text{seq}}_{\tau+1}(\boldsymbol{x})$ for all $\boldsymbol{x} \in D$ and $\tau \in \mathcal{S}$, then $f(\boldsymbol{x}) \in C^{\text{batch}}_t(\boldsymbol{x})$ for all $\boldsymbol{x} \in D$ and all $t \geq 1$ because every t has an image in $\mathcal{S}$. Thus $f(\boldsymbol{x}) \in C^{\text{seq}}_{\tau+1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall \tau \in \mathcal{S} \implies f(\boldsymbol{x}) \in C^{\text{batch}}_t(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1$. The lemma follows because if the probability of the sufficient condition is at least $1 - \delta$, then the probability of the implied condition must also be at least $1 - \delta$. ■

The high-probability confidence intervals are next related to the instantaneous regret and thence to the cumulative regret. We first state several supporting lemmas.

Lemma 10 *(From Lemma 5.2 of Srinivas et al., 2012) If $f(\boldsymbol{x}) \in C^{batch}_t(\boldsymbol{x})$ for all $\boldsymbol{x} \in D$ and all $t \geq 1$, when actions are selected via Equation (7), $r_t \leq 2\beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t)$, $\forall t \geq 1$.*

Proof By Equation (7), $\boldsymbol{x}_t$ is chosen at each time t such that $\mu_{\text{fb}[t]}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}) \leq \mu_{\text{fb}[t]}(\boldsymbol{x}_t) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t), \forall \boldsymbol{x} \in D$, including for any optimal choice $\boldsymbol{x} = \boldsymbol{x}^*$. Since the instantaneous regret is defined as $r_t = f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t)$ and by assumption both $f(\boldsymbol{x}^*)$ and $f(\boldsymbol{x}_t)$ are contained within their respective confidence intervals,

$$\begin{aligned} r_t &\leq [\mu_{\text{fb}[t]}(\boldsymbol{x}^*) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}^*)] - [\mu_{\text{fb}[t]}(\boldsymbol{x}_t) - \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t)] \\ &\leq [\mu_{\text{fb}[t]}(\boldsymbol{x}_t) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t)] - [\mu_{\text{fb}[t]}(\boldsymbol{x}_t) - \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t)] \\ &\leq 2\beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t). \end{aligned}$$

■

Lemma 11 *(Lemma 5.3 of Srinivas et al., 2012) The mutual information gain with respect to f for the actions selected, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, can be expressed in terms of the predictive variances as*

$$I(f; \boldsymbol{y}_{1:T}) = \frac{1}{2}\sum_{t=1}^{T} \log(1 + \sigma_n^{-2}\sigma^2_{t-1}(\boldsymbol{x}_t)).$$

This statement is a result of the additivity of the conditional mutual information gain of observations of a Gaussian.

Lemma 12 *(Extension of Lemma 5.4 of Srinivas et al., 2012) Let $k(\boldsymbol{x}, \boldsymbol{x}) \leq 1, \forall \boldsymbol{x} \in D$. If $f(\boldsymbol{x}) \in C_t^{batch}(\boldsymbol{x})$, $\forall \boldsymbol{x} \in D$, $\forall t \geq 1$, and given that actions $\boldsymbol{x}_t$, $\forall t \in \{1, \ldots, T\}$ are selected using Equation (7), it holds that*

$$R_T \leq \sqrt{TC_1\beta_T\gamma_T},$$

where $C_1 = 8/\log(1 + \sigma_n^{-2})$, γ_T is defined in Equation (4), and β_t is defined in Equation (19).

Proof Given $f(\boldsymbol{x}) \in C_t^{\text{batch}}(\boldsymbol{x})$, $\forall \boldsymbol{x} \in D$, $\forall t \geq 1$, Lemma 10 bounds the instantaneous regret r_t as $r_t \leq 2\beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t)$, $\forall t \geq 1$. The square of the right-hand quantity may be manipulated algebraically to show that $4\beta_t\sigma_{t-1}^2(\boldsymbol{x}_t) \leq C_1\beta_t[1/2\log(1 + \sigma_n^{-2}\sigma_{t-1}^2(\boldsymbol{x}_t))]$. This manipulation exploits the facts that $\sigma_{t-1}^2(\boldsymbol{x}) \leq k(\boldsymbol{x}, \boldsymbol{x}) \leq 1, \forall \boldsymbol{x} \in D$ and that $x/\log(1 + x)$ is non-decreasing for $x \in [0, \infty)$. Summing in time and noting that β_t is non-decreasing, $\sum_{t=1}^T 4\beta_t\sigma_{t-1}^2(\boldsymbol{x}_t) \leq C_1\beta_T\sum_{t=1}^T 1/2\log(1 + \sigma_n^{-2}\sigma_{t-1}^2(\boldsymbol{x}_t)) = C_1\beta_T I(f; \boldsymbol{y}_{1:T})$ by Lemma 11. Thus, by Equation (4), $\sum_{t=1}^T r_t^2 \leq C_1\beta_T\gamma_T$. The claim then follows as a consequence of the Cauchy-Schwarz inequality, since $R_T^2 \leq T\sum_{t=1}^T r_t^2$. ■

Proof [Proof of Theorem 2, Case 1] Taken together, Lemmas 8 through 12, a bound C satisfying Equation (18), and a high-probability guarantee that some set of sequential confidence intervals always contain the values of f allow us to construct a batch algorithm with high-probability regret bounds. Lemma 7 gives us precisely such a guarantee in the case that D is of finite size. Employing Lemma 7 and Lemma 12, and noting that the result holds for all $T \geq 1$, Case 1 of Theorem 2 follows as an immediate corollary. ■

A.2 Case 2: $D \subset \mathbb{R}^d$

Case 2 of Theorem 2 deals with decision sets which are continuous regions of $\mathbb{R}^d$. As a note, we assume that it is possible to select $\boldsymbol{x}_t \in D$ according to Equation (7), i.e., as the maximizer of the decision rule over D. This assumption is non-trivial in practice; this is a non-convex optimization problem in general, though of a function which is perhaps not too ill-behaved, e.g., it is differentiable under the assumptions of Case 2. Nevertheless, we make this assumption and proceed with our analysis.

In the proof of Lemma 7, $P(\exists \boldsymbol{x} \in D : |f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})| > \beta_t^1/2\sigma_{t-1}(\boldsymbol{x}))$ is bounded via a union bound over D as at most $|D|\exp(-\alpha_t/2)$. Unfortunately, since this bound scales directly with the number of elements in D, this is not useful when D is continuous. We instead use a very similar analysis to establish high-probability confidence intervals on a *subset* D_t of D; using a high-probability bound on the derivatives of the sample paths drawn from the GP, we then proceed to upper-bound $f(\boldsymbol{x})$ for $\boldsymbol{x} \in D \setminus D_t$. Next, we establish a high-probability guarantee for the containment of the reward corresponding to the actions actually taken within their respective confidence bounds at any time, and combine these results to bound the regret r_t suffered in round t. With some slight modifications and careful choices of the scaling of D_t and β_t, the remainder of the analysis from Case 1 can be employed to establish the required bound on R_T for all $T \geq 1$.

Lemma 13 *(From Lemma 5.6 of Srinivas et al., 2012) Specify a discrete set $D_t \subset D$ for every $t \geq 1$, where $D \subseteq [0,l]^d$ and $|D_t|$ is finite. Also specify $\delta \in (0,1)$ and let $\beta_t = 2\exp(2C)\log(|D_t|\pi_t/\delta)$, where $\pi_t > 0, \forall t \geq 1$, $\sum_{t=1}^{\infty} \pi_t^{-1} = 1$, and $I(f(\boldsymbol{x}); \boldsymbol{y}_{fb[t]+1:t-1}|\boldsymbol{y}_{1:t}) \leq C, \forall \boldsymbol{x} \in D, \forall t \geq 1$. Then,*

$$P(|f(\boldsymbol{x}) - \mu_{fb[t]}(\boldsymbol{x})| \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D_t, \forall t \geq 1) \geq 1 - \delta.$$

Proof The proof of this lemma is very similar to that of Lemma 7. First, conditioned on actions $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{\text{fb}[t]}\}$ and observations $\{y_1, y_2, \ldots, y_{\text{fb}[t]}\}$, $f(\boldsymbol{x}) \sim \mathcal{N}(\mu_{\text{fb}[t]}(\boldsymbol{x}), \sigma^2_{\text{fb}[t]}(\boldsymbol{x}))$; thus, $(f(\boldsymbol{x}) - \mu_{\text{fb}[t]}(\boldsymbol{x}))/\sigma_{\text{fb}[t]}(\boldsymbol{x}) \sim \mathcal{N}(0,1)$. If $a \sim \mathcal{N}(0,1)$, $c \geq 0$, then $P(a > c) \leq 1/2\exp(-c^2/2)$. Using this inequality and a union bound over all $\boldsymbol{x} \in D_t$, we obtain the following result for general $\beta_t > 0$:

$$\begin{aligned} P\left(\frac{|f(\boldsymbol{x}) - \mu_{\text{fb}[t]}(\boldsymbol{x})|}{\sigma_{\text{fb}[t]}(\boldsymbol{x})} \leq \beta_t^{1/2}\frac{\sigma_{t-1}(\boldsymbol{x})}{\sigma_{\text{fb}[t]}}, \forall \boldsymbol{x} \in D_t\right) &\geq 1 - |D_t|\exp\left(-\frac{\beta_t}{2}\frac{\sigma^2_{t-1}(\boldsymbol{x})}{\sigma^2_{\text{fb}[t]}}\right) \\ &\geq 1 - |D_t|\exp(-\beta_t\exp(-2C)/2). \end{aligned}$$

Analogous to the proof of Lemma 7, let $\delta/\pi_t = |D_t|\exp(-\beta_t\exp(-2C)/2)$, implicitly defining β_t, and union bound in time, yielding the desired result. ■

Note that if at each time $t \geq 1$ we specify a particular $\mathbf{z}_t \in D$ and choose $D_t = \mathbf{z}_t$, Lemma 13 implies that $P(|f(\mathbf{z}_t) - \mu_{\text{fb}[t]}(\mathbf{z}_t)| \leq \tilde{\beta}_t^{1/2}\sigma_{t-1}(\mathbf{z}_t), \forall t \geq 1) \geq 1 - \delta$, where $\tilde{\beta}_t \geq 2\exp(2C)\log(\pi_t/\delta)$. This fact will be employed for $\mathbf{z}_t = \boldsymbol{x}_t$ below.

Next, we upper bound the value of $f(\boldsymbol{x}^*)$, where $\boldsymbol{x}^*$ is possibly within $D \setminus D_t$. Let $[\boldsymbol{x}]_t = \text{argmin}_{\boldsymbol{x}' \in D_t} ||\boldsymbol{x} - \boldsymbol{x}'||_1$, i.e., the closest point in D_t to $\boldsymbol{x}$, in the sense of 1-norm. As a technical point, $[\boldsymbol{x}]_t$ may not be uniquely determined; any 1-norm minimizing element of D_t is sufficient for the following discussion.

Lemma 14 *(From Lemma 5.7 of Srinivas et al., 2012) Specify $\delta \in (0,1)$ and let τ_t be a time-varying parameter. Let $D_t \subset D$ be chosen such that $||\boldsymbol{x} - [\boldsymbol{x}]_t||_1 \leq ld/\tau_t, \forall \boldsymbol{x} \in D, \forall t \geq 1$. Let the statement*

$$P(\sup_{\boldsymbol{x} \in D} |\partial f(\boldsymbol{x})/\partial \boldsymbol{x}_i| < L, \forall i \in \{1, \ldots, d\}) \geq 1 - da\exp(-L^2/b^2)$$

hold for any $L > 0$ for some corresponding $a \geq \delta/(2d)$, $b > 0$, where $\boldsymbol{x}_i$ denotes the ith dimension of $\boldsymbol{x}$. Choose $L = b\sqrt{\log(2da/\delta)}$, $\tau_t = dt^2bl\sqrt{\log(2da/\delta)}$, and $\beta_t \geq 2\exp(2C) \cdot \log(2|D_t|\pi_t/\delta)$, where $\pi_t > 0, \forall t \geq 1$, $\sum_{t=1}^{\infty} \pi_t^{-1} = 1$, and $I(f(\boldsymbol{x}); \boldsymbol{y}_{fb[t]+1:t-1}|\boldsymbol{y}_{1:t}) \leq C, \forall \boldsymbol{x} \in D, \forall t \geq 1$. Then,

$$P\left(|f(\boldsymbol{x}^*) - \mu_{fb[t]}([\boldsymbol{x}^*]_t)| < \beta_t^{1/2}\sigma_{t-1}([\boldsymbol{x}^*]_t) + \frac{1}{t^2}, \forall t \geq 1\right) \geq 1 - \delta$$

Proof For the specified choice of L, we obtain $P(|\partial f(\boldsymbol{x})/\partial \boldsymbol{x}_i| < b\sqrt{\log(2da/\delta)}, \forall \boldsymbol{x} \in D, \forall i \in \{1, \ldots, d\}) \geq 1 - \delta/2$. Thus, with probability $\geq 1 - \delta/2$,

$$\begin{aligned} |f(\boldsymbol{x}^*) - f([\boldsymbol{x}^*]_t)| &\leq b\sqrt{\log(2da/\delta)} \cdot ||\boldsymbol{x}^* - [\boldsymbol{x}^*]_t||_1 \\ &\leq b\sqrt{\log(2da/\delta)} \cdot ld/\tau_t \\ &\leq \frac{1}{t^2}. \end{aligned}$$

By Lemma 13, for β_t as chosen above, $|f([\boldsymbol{x}^*]_t) - \mu_{\text{fb}[t]}([\boldsymbol{x}^*]_t)| \leq \beta_t^{1/2}\sigma_{t-1}([\boldsymbol{x}^*]_t), \forall t \geq 1$ with probability $\geq 1 - \delta/2$. The result follows by a union bound on the possible failures these two events. ■

Note that Lemma 14 states that if we know the size of a suitable discretization D_t of D, we may choose β_t such that we may establish a high probability upper bound on f over all of D. Note also that a larger β_t is acceptable and that D_t itself is not required to prove the result. Our next result proves the existence and size of a sufficient discretization of D; we will then choose β_t according to this provably existent discretization and entirely avoid its explicit construction.

In the following lemma, $\lceil \tau_t \rceil$ denotes the smallest integer which is at least τ_t.

Lemma 15 *There exists a discretization D_t of $D \subseteq [0, l]^d$, D compact and convex, where $|D_t| \leq \lceil \tau_t \rceil^d$ and $||\boldsymbol{x} - [\boldsymbol{x}]_t||_1 \leq \frac{ld}{\tau_t}, \forall \boldsymbol{x} \in D$.*

Proof It is sufficient to construct an example discretization. One way to do so is to first generate an ϵ-cover of $[0, l]^d \supseteq D$, where $\epsilon = ld/2\tau_t$; this can be done by placing a ball centered at each location in $\{\frac{l}{2\lceil \tau_t \rceil}, \frac{3l}{2\lceil \tau_t \rceil}, \ldots, \frac{(2\lceil \tau_t \rceil - 1)l}{2\lceil \tau_t \rceil}\}^d$, such that each point in $[0, l]^d$ (and thus D) is at most $ld/2\lceil \tau_t \rceil \leq ld/2\tau_t$ from the nearest point in this set (i.e., is within at least one of the closed balls) and every ball center lies within $[0, l]^d$. Denote this set of ball centers $\mathcal{A}$ and note that $|\mathcal{A}| = \lceil \tau_t \rceil^d$. For each $\boldsymbol{x} \in \mathcal{A}$, denote the corresponding 1-norm ball of radius $ld/2\tau_t$ as $B^1_{ld/2\tau_t}(\boldsymbol{x})$. We now use $\mathcal{A}$ to construct D_t, such D_t is an ϵ-cover for D, for $\epsilon = ld/\tau_t$. Begin with D_t empty and iterate over $\boldsymbol{x} \in \mathcal{A}$. If $\boldsymbol{x} \in D$, add it to D_t. If $\boldsymbol{x} \notin D$, but $B^1_{ld/2\tau_t}(\boldsymbol{x}) \cap D \neq \emptyset$, add any point in this intersection to D_t. If $\boldsymbol{x} \notin D$ and $B^1_{ld/2\tau_t}(\boldsymbol{x}) \cap D = \emptyset$, do not add $\boldsymbol{x}$ to D_t. By construction, $\boldsymbol{x} \in D_t \implies \boldsymbol{x} \in D$. Since the triangle inequality implies $B^1_{ld/2\tau_t}(\boldsymbol{x}) \subset B^1_{ld/\tau_t}(\boldsymbol{x}'), \forall \boldsymbol{x}' \in B^1_{ld/2\tau_t}(\boldsymbol{x})$, we have the result that $\bigcup_{\boldsymbol{x}' \in D_t} B^1_{ld/\tau_t}(\boldsymbol{x}') \supseteq \bigcup_{\boldsymbol{x}: \boldsymbol{x} \in \mathcal{A}, B^1_{ld/2\tau_t}(\boldsymbol{x}) \cap D \neq \emptyset} B^1_{ld/2\tau_t}(\boldsymbol{x})$, where this second union is by definition a cover for D. D_t is thus an ϵ-cover for D for $\epsilon = ld/\tau_t$ and therefore a satisfactory discretization of size no more than $\lceil \tau_t \rceil^d$. ■

Lemma 14 also rests on a bound on the derivatives of $f(\boldsymbol{x})$ with respect to $\boldsymbol{x}_i, \forall i = 1, \ldots, d$. Such a bound can be created if the kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$ defining the distribution over f is sufficiently many times differentiable with respect to $\boldsymbol{x}$ and $\boldsymbol{x}'$.

Lemma 16 *(From Srinivas et al., 2012, Appendix I) If $f \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$ and derivatives of $k(\boldsymbol{x}, \boldsymbol{x}')$ exist up to fourth order with respect to $\boldsymbol{x}, \boldsymbol{x}' \in D$, then f is almost surely continuously differentiable and there exist positive constants a, b, and L, such that*

$$P(|f(\boldsymbol{x}) - f(\boldsymbol{x}')| \leq L||\boldsymbol{x} - \boldsymbol{x}'||_1, \forall \boldsymbol{x}, \boldsymbol{x}' \in D) \geq 1 - \delta,$$

where $L = b\sqrt{\log(da/\delta)}$.

Proof If all fourth order partial derivatives of $k(\boldsymbol{x}, \boldsymbol{x}')$ exist, the derivatives of f are themselves Gaussian processes with a kernel function corresponding to the twice differentiated k and there exist positive constants $a, \{b_1, \ldots, b_d\}$ such that $P(\sup_{\boldsymbol{x} \in D} |\partial f / \partial \boldsymbol{x}_j| > L_1) \leq$

$a\exp(-b_jL_1^2), \forall j \in \{1,\ldots,d\}$, for any $L_1 > 0$ (Theorem 5 of Ghosal and Roy, 2006). Let $L_1 = \sqrt{\log(da/\delta)}/(\min_j \sqrt{b_j})$. Note that a can be chosen arbitrarily large and $a \geq \delta/d$ (required so the argument of the logarithm is ≥ 1) implies $(da/\delta)^{(-b_j/(\min_j b_j))} \leq \delta/da$; union bounding in j, we thus obtain that $P(\sup_{\boldsymbol{x}\in D} |\partial f/\partial \boldsymbol{x}_j| > L_1, \forall j \in \{1,\ldots,d\}) \leq \delta$. Reparameterizing, choose $b \geq \max_j b_j^{-1/2} > 0$, and define $L = b\sqrt{\log(da/\delta)}$. Using this bound on the supremum of the derivatives of f, and a piecewise construction of the path from $\boldsymbol{x}$ to $\boldsymbol{x}'$ according to the unit vectors, the result follows. ■

In the proof of Lemma 15, we bound the size of the virtual decision set D_t as $\lceil \tau_t \rceil^d$. We can instead use τ_t^d if τ_t is an integer. Luckily, we can always make a and b bigger, e.g., such that $bl\sqrt{\log(da/\delta)}$ is an integer. If this quantity is an integer, so is $\tau_t = dt^2 bl\sqrt{\log(da/\delta)}$.

Next, we bound the regret r_t incurred at round t.

Lemma 17 *(From Lemma 5.8 of Srinivas et al., 2012) Specify $\delta \in (0,1)$. Let the statement*

$$P(\sup_{\boldsymbol{x}\in D} |\partial f(\boldsymbol{x})/\partial \boldsymbol{x}_i| < L, \forall i \in \{1,\ldots,d\}) \geq 1 - da\exp(-L^2/b^2)$$

hold for any $L > 0$ and positive constants a, b. Choose a, b such that $a \geq \delta/(4d)$ and $bl\sqrt{\log(4da/\delta)}$ is an integer. Let actions be selected using Equation (7), where

$$\beta_t = 2\exp(2C)[\log(4\pi_t/\delta) + d\log(dt^2 bl\sqrt{\log(4da/\delta)})],$$

$\pi_t > 0, \forall t \geq 1$, $\sum_{t=1}^{\infty} \pi_t^{-1} = 1$, and $I(f(\boldsymbol{x}); \boldsymbol{y}_{fb[t]+1:t-1} | \boldsymbol{y}_{1:t}) \leq C, \forall \boldsymbol{x} \in D, \forall t \geq 1$. Then,

$$P(r_t \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}) + t^{-2}, \forall t \geq 1) \geq 1 - \delta.$$

Proof By Lemma 15, for any round t, we may construct a discretization D_t of size no more than $\lceil \tau_t \rceil^d$, such that $||\boldsymbol{x} - [\boldsymbol{x}]_t||_1 \leq ld/\tau_t$, where $\tau_t = dt^2 bl\sqrt{\log(4da/\delta)}$ and τ_t is an integer. Choosing β_t as specified and implicitly constructing such a D_t, by Lemma 14, it follows that $P(|f(\boldsymbol{x}^*) - \mu([\boldsymbol{x}^*]_{\text{fb}[t]})| < \beta_t^{1/2}\sigma_{t-1}([\boldsymbol{x}^*]_t) + t^{-2}) \geq 1 - \delta/2$. Applying Lemma 13 to every $\boldsymbol{x}_t$, $t \geq 1$, for any $\tilde{\beta}_t \geq 2\exp(2C)\log(2\pi_t/\delta)$, $P(|f(\boldsymbol{x}_t) - \mu_{\text{fb}[t]}(\boldsymbol{x}_t)| \leq \tilde{\beta}_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t) \forall t \geq 1) \geq 1 - \delta/2$. As specified, $\beta_t \geq 2\exp(2C)\log(2\pi_t/\delta)$, and so $f(\boldsymbol{x}_t)$ is bounded with probability $\geq 1 - \delta/2$. By Equation (7), $\mu_{\text{fb}[t]}(\boldsymbol{x}_t) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t) \geq \mu_{\text{fb}[t]}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1$, and so,

$$\begin{aligned}
r_t &= f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t) \\
&\leq [\mu_{\text{fb}[t]}([\boldsymbol{x}^*]_t) + \beta_t^{1/2}\sigma_{t-1}([\boldsymbol{x}^*]_t) + t^{-2}] - [\mu_{\text{fb}[t]}(\boldsymbol{x}_t) - \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t)] \\
&\leq [\mu_{\text{fb}[t]}(\boldsymbol{x}_t) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t) + t^{-2}] - [\mu_{\text{fb}[t]}(\boldsymbol{x}_t) - \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t)] \\
&= 2\beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t) + t^{-2},
\end{aligned}$$

with probability $\geq 1 - \delta$. ■

Proof [Proof of Theorem 2, Case 2] By Lemma 17, for $\beta_t = 2\exp(2C)[\log(4\pi_t/\delta) + d\log(dt^2 bl\sqrt{\log(4da/\delta)})]$, it follows that $r_t \leq 2\beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t) + t^{-2}, \forall t \geq 1$, with probability

$\geq 1-\delta$. Consequently,

$$R_T = \sum_{t=1}^{T} r_t \leq \sum_{t=1}^{T} 2\beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t) + \sum_{t=1}^{T} t^{-2}$$
$$\leq \sqrt{TC_1\beta_T\gamma_T} + \pi^2/6,$$

for all $T \geq 1$, with probability $\geq 1-\delta$, where $C_1 = 8/\log(1+\sigma_n^{-2})$ and the second inequality follows via the argument advanced in the proof of Lemma 12 in Case 1; this argument uses the information gain and Lemma 11 to bound the sum of the squares of the terms, and then the Cauchy-Schwarz inequality to bound the original sum. Theorem 2, Case 2 follows. ■

A natural extension is to the case where the GP prior mean is non-zero, but is known and Lipschitz-continuous. This is straight-forward, since bounds on $\sup_{\boldsymbol{x}\in D} |\partial f(\boldsymbol{x})/\partial \boldsymbol{x}_i|$ or the generalization error in the prior mean may be naturally obtained.

A.3 Case 3: Finite RKHS Norm of f

Case 3 involves a reward function f with a finite RKHS norm with respect to the algorithm's GP prior. Fortunately, Srinivas et al. (2012) again have a result which creates confidence intervals in this situation.

Lemma 18 *(Theorem 6 from Srinivas et al., 2012) Specify $\delta \in (0,1)$. Let $k(\boldsymbol{x},\boldsymbol{x}')$ be an assumed Gaussian process kernel and let $||f||_k$ be the RKHS norm of f with respect to k. Let k be such that $k(\boldsymbol{x},\boldsymbol{x}) \leq 1$ for all $\boldsymbol{x} \in D$. Assume noise variables ϵ_t are from a martingale difference sequence, such that they are uniformly bounded by σ_n. Define*

$$\alpha_t = 2M + 300\gamma_t \log^3(t/\delta),$$

where $M \geq ||f||_k^2$. Then

$$P(|f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})| \leq \alpha_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1) \geq 1-\delta.$$

In brief, the reproducing property of kernels implies that the corresponding inner product of $g(\boldsymbol{x})$ with $k(\boldsymbol{x},\boldsymbol{x}')$, denoted $\langle g(\boldsymbol{x}), k(\boldsymbol{x},\boldsymbol{x}')\rangle_k$, is $\langle g(\boldsymbol{x}), k(\boldsymbol{x},\boldsymbol{x}')\rangle_k = g(\boldsymbol{x}')$ for any kernel k, and so, using such an inner product and the Cauchy-Schwarz inequality,

$$|\mu_t(\boldsymbol{x}) - f(\boldsymbol{x})| = \sqrt{\langle \mu_t(\boldsymbol{x}') - f(\boldsymbol{x}'), k_t(\boldsymbol{x},\boldsymbol{x}')\rangle_{k_t}^2}$$
$$\leq \sqrt{\langle k_t(\boldsymbol{x},\boldsymbol{x}'), k_t(\boldsymbol{x},\boldsymbol{x}')\rangle_{k_t}\langle \mu_t(\boldsymbol{x}) - f(\boldsymbol{x}), \mu_t(\boldsymbol{x}) - f(\boldsymbol{x})\rangle_{k_t}}$$
$$= \sqrt{k_t(\boldsymbol{x},\boldsymbol{x})}||\mu_t - f||_{k_t} = \sigma_t(\boldsymbol{x})||\mu_t - f||_{k_t},$$

where k_t is the posterior kernel and μ_t is the posterior mean at time t. This implies that an upper bound on $||\mu_t - f||_{k_t}$ gives an upper bound on the regret which can be incurred at the selection of action $t+1$ (via the argument of Lemma 10), suggesting that our UCB width multiplier $\alpha_t^{1/2}$ should have a form related to the growth of $||\mu_t - f||_{k_t}$. Since $||g||_{k_t}^2 = ||g||_k^2 + \sigma_n^{-2}\sum_{\tau=1}^{t} g(\boldsymbol{x}_\tau)^2$ for a function g, we may substitute $g = \mu_t - f$ and

then factor the squared norm. We thus obtain a term in $||f||_k^2$ and a second term in the observations and noise terms y_t and ϵ_t. We assume we have a bound $M \geq ||f||_k^2$, and so our problem reduces to choosing an α_t sufficiently large to surpass the growth of the second term. Through an inductive argument on the probability of confidence interval failure up to the tth action and using the α_t chosen, Srinivas et al. (2012) show that these confidence intervals hold with probability at least $1 - \delta$. Importantly, this argument does *not* use the decision-making process of the algorithm, instead only relying on the algorithm's internal GP posterior over f and the characteristics of the martingale noise process. We refer the interested reader to Appendix II of Srinivas et al. (2012) for the details.[5]

In terms of proving Case 3 of Theorem 2, Lemma 18 is just the sort of statement we need; it establishes a precisely analogous result to Lemma 7, providing a set of confidence intervals $C_\tau^{\text{seq}}(\boldsymbol{x})$ such that for $\tau = \text{fb}[t] + 1$, we can construct β_t and thus $C_t^{\text{batch}}(\boldsymbol{x})$ such that $C_t^{\text{batch}}(\boldsymbol{x}) \supseteq C_{\text{fb}[t]+1}^{\text{seq}}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1$.

Proof [Proof of Theorem 2, Case 3] By Lemma 18, for the specified value of δ, and choosing $\alpha_t = 2M + 300\gamma_t \log^3(t/\delta)$, $P(f(\boldsymbol{x}) \in C_\tau^{\text{seq}}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall \tau \geq 1) \geq 1 - \delta$. Let $\beta_t = \exp(2C)\alpha_{\text{fb}[t]+1}$, where C satisfies Equation (18); by Lemmas 8 and 9, $P(f(\boldsymbol{x}) \in C_t^{\text{batch}}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1) \geq 1 - \delta$. Then, by Lemma 10, the instantaneous regret is bounded as $r_t \leq \beta_t^{1/2}\sigma_{t-1}^2(\boldsymbol{x})$ for each $t \geq 1$. Application of Lemmas 11 and 12 yields the result. ■

Appendix B. Initialization Set Size Bounds

Thorough initialization of GP-BUCB can drive down the constant C, which bounds the information which can be hallucinated in the course of post-initialization batches and also governs the asymptotic scaling of the regret bound with batch size B. First, we connect the information which can be gained in post-initialization batches with the amount of information being gained in the initialization, through Lemma 4, the formal statement of which is in Section 4.5, and the proof of which is presented here.

Proof [Proof of Lemma 4] Since the initialization procedure is greedy, the marginal information gain $1/2 \log(1 + \sigma_n^{-2}\sigma_{t-1}^2(\boldsymbol{x}))$ is a monotonic function of $\sigma_{t-1}(\boldsymbol{x})$, and information gain is submodular (See Section 3.3), the information gain from $\boldsymbol{y}_{T^{\text{init}}}$, which corresponds to $\boldsymbol{x}_{T^{\text{init}}}^{\text{init}}$, the last element of the initialization set, must be the smallest marginal information gain in the initialization process, and thus no greater than the mean information gain, i.e.,

$$I\left(f; \boldsymbol{y}_{T^{\text{init}}} \mid \boldsymbol{y}_{D_{T^{\text{init}}-1}^{\text{init}}}\right) \leq I\left(f; \boldsymbol{y}_{D_{T^{\text{init}}}^{\text{init}}}\right) / T^{\text{init}}.$$

5. In the proof of Lemma 7.2 of Srinivas et al. (2012), the (GP-UCB-type) *algorithm's internal* GP posterior model of f is exploited to examine $||\mu_t - f||_{k_t}$ in terms of the model's individual conditional distributions for $y_\tau, \tau = 1, \ldots, t$. This argument relies on the GP model and its assumption of i.i.d. Gaussian noise, but does *not* change or violate the problem assumption that the actual observation noise ϵ_t is from an arbitrary, uniformly bounded martingale difference sequence.

Further, again because information gain is submodular and the initialization set was constructed greedily, no subsequent action can yield greater marginal information gain. Thus,

$$\gamma^{\text{init}}_{B-1} \leq (B-1) \cdot I\left(f; \boldsymbol{y}_{T^{\text{init}}} \mid \boldsymbol{y}_{D^{\text{init}}_{T^{\text{init}}-1}}\right).$$

Combining these two inequalities with the definition of $\gamma_{T^{\text{init}}}$ yields the result. ■

Next, we examine how Lemma 4 can be used to bound the regret of the two-stage algorithm. In the two stage algorithm, we may consider two sets of confidence intervals, which do not coincide during the construction of the initialization set, and do coincide afterward; specifically, let $\tilde{\text{fb}}[t]$ be a virtual feedback mapping which receives feedback at every point the actual feedback mapping $\text{fb}[t]$ does, i.e., at time T^{init} and at times thereafter such that Equation (10) is satisfied for all $t \geq T^{\text{init}}+1$ for $C = 1/2\log(C')$, and in addition, *receives feedback during the construction of the initialization set* such that Equation (10) is also satisfied during this time. While the virtual feedback mapping $\tilde{\text{fb}}[t]$ is of course not used by the algorithm to construct confidence intervals or make decisions, it will prove useful for our proof.

Proof [Theorem 5] Assume that there can be constructed an initialization set D^{init} of size T^{init}, subsequent to which the information gain of any batch selected by the GP-BUCB decision rule with respect to $f(\boldsymbol{x})$ for any $\boldsymbol{x} \in D$ is no more than C. Then, for the values of β_t given in the statement of Theorem 2 and using $C = 1/2\log(C')$, where C' is dictated by the choice of kernel,

$$P(|f(\boldsymbol{x}) - \mu_{\tilde{\text{fb}}[t]}(\boldsymbol{x})| \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1) \geq 1-\delta$$

in Cases 1 & 3, and the corresponding statement with $|f(\boldsymbol{x}) - \mu_{\tilde{\text{fb}}[t]}(\boldsymbol{x})| \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}) + t^{-2}$ in Case 2. This result follows from the following sets of lemmas: 7, 8, and 9 (Case 1); 13, 14, 15, and 16 (Case 2); and 18, 8, and 9 (Case 3). Since the actual feedback mapping $\text{fb}[t]$ and $\tilde{\text{fb}}[t]$ coincide for $t \geq T^{\text{init}}+1$, the virtual feedback mapping's probability of confidence interval containment (correctness to known error in Case 2) at all times $t \geq 1$ is a lower bound on the probability that the true, post-initialization confidence intervals constructed using $\text{fb}[t]$ are correct (correct to known error in Case 2); i.e.,

$$\begin{aligned} P(|f(\boldsymbol{x}) - \mu_{\text{fb}[t]}(\boldsymbol{x})| &\leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq T^{\text{init}}+1) \\ &\geq P(|f(\boldsymbol{x}) - \mu_{\tilde{\text{fb}}[t]}(\boldsymbol{x})| \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \forall \boldsymbol{x} \in D, \forall t \geq 1) \\ &\geq 1-\delta \end{aligned}$$

in Cases 1 & 3 and similarly,

$$P(|f(\boldsymbol{x}) - \mu_{\text{fb}[t]}(\boldsymbol{x})| \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}) + t^{-2}, \forall \boldsymbol{x} \in D, \forall t \geq T^{\text{init}}+1) \geq 1-\delta$$

in Case 2. By Lemma 11, $I(\boldsymbol{y}_{T^{\text{init}}+1:T}; \{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_T)\}) = \frac{1}{2}\sum_{t=T^{\text{init}}+1}^{T} \log(1+\sigma_n^{-2}\sigma_{t-1}^2(\boldsymbol{x}_t))$. Define $R_{T^{\text{init}}+1:T} = \sum_{t=T^{\text{init}}+1}^{T} r_t$. By the same Cauchy-Schwarz argument used in each proof in Appendix A, with probability $\geq 1-\delta$,

$$R_{T^{\text{init}}+1:T} \leq \sqrt{(T-T^{\text{init}})C_1\beta_T\gamma^{\text{init}}_{(T-T^{\text{init}})}} \leq \sqrt{TC_1\beta_T\gamma_T},$$

for all $T \geq 1$, in Cases 1 & 3. The second inequality in the above argument is wasteful, but simplifies the statement of the theorem without affecting asymptotic scaling. In Case 2, we must amend the final bound by adding a term $\pi^2/6$ to the right-hand side as an upper bound on $\sum_{t=T^{\text{init}}+1}^{T} t^{-2}$, itself bounding the generalization error from the virtual discretization. Noting that $R_T = R_{T^{\text{init}}} + R_{T^{\text{init}}+1:T}$, $R_{T^{\text{init}}} = \sum_{t=1}^{T^{\text{init}}} f(x^*) - f(x_t) \leq 2T^{\text{init}}||f||_\infty$, $C' \geq 1$, and $\beta_t = (C')^2 \alpha_{\text{fb}[t]+1}$ (α_t in Case 2), the result follows. ■

B.1 Initialization Set Size: Linear Kernel

For the linear kernel, there exists a logarithmic bound on the maximum information gain of a set of queries, precisely, $\exists\, \eta \geq 0 : \gamma_t \leq \eta d \log(t+1)$ (Srinivas et al., 2010). We attempt to initialize GP-BUCB with a set D^{init} of size T^{init}, where, motivated by this bound and the form of Inequality (15), we assume T^{init} is of the form

$$T^{\text{init}} = k\eta d(B-1)\log B. \tag{20}$$

We must show that there exists a k of finite size for which an initialization set of size T^{init}, as in Equation (20), implies that any subsequent set $\mathcal{S}$, $|\mathcal{S}| = B-1$, produces a conditional information gain with respect to f of no more than C. This requires showing that the inequality $\frac{B-1}{T^{\text{init}}}\gamma_{T^{\text{init}}} \leq C$ holds for this choice of k and thus T^{init}. Since we consider non-trivial batches, i.e., $B-1 \geq 1$, if k is large enough that $k\eta d(B-1) \geq 1$,

$$\log(\log(B) + 1/(k\eta d(B-1))) \leq \log(\log(B)+1) \leq \log B.$$

Using Equation (20) and the bound for $\gamma_{T^{\text{init}}}$, and following algebraic rearrangement, this inequality implies that if $k\eta d(B-1) \geq 1$,

$$\frac{B-1}{T^{\text{init}}}\gamma_{T^{\text{init}}} \leq C \Longleftarrow \frac{\log k}{k\log B} + \frac{\log\eta + \log d}{k\log B} + \frac{2}{k} \leq C.$$

By noting that the maximum of $\frac{\log k}{k}$ over $k \in (0,\infty)$ is $1/e$ and choosing for convenience $C = 2/e$, we obtain for $k \geq 1/(\eta d(B-1))$:

$$\frac{B-1}{T^{\text{init}}}\gamma_{T^{\text{init}}} \leq \frac{2}{e} \Longleftarrow \frac{1}{e\log B} + \frac{1}{k}\left(\frac{\log\eta + \log d + 2\log B}{\log B}\right) \leq \frac{2}{e}.$$

Choosing k to satisfy both constraints simultaneously,

$$\frac{B-1}{T^{\text{init}}}\gamma_{T^{\text{init}}} \leq \frac{2}{e} \Longleftarrow k \geq \max\left[\frac{1}{\eta d(B-1)}, \frac{e(\log\eta + \log d + 2\log B)}{2\log(B) - 1}\right].$$

Thus, for a linear kernel and such a k, an initialization set D^{init} of size T^{init}, where $T^{\text{init}} \geq k\eta d(B-1)\log(B)$, ensures that the hallucinated conditional information in any future batch of size B is $\leq \frac{2}{e}$.

B.2 Initialization Set Size: Matérn Kernel

For the Matérn kernel, $\gamma_t \leq \nu t^{\epsilon}$, $\epsilon \in (0,1)$ for some $\nu > 0$ (Srinivas et al., 2010). Hence:

$$\frac{(B-1)}{T^{\text{init}}}\gamma_{T^{\text{init}}} \leq C \Longleftarrow \frac{\nu(B-1)(T^{\text{init}})^{\epsilon}}{T^{\text{init}}} = \nu(B-1)(T^{\text{init}})^{\epsilon-1} \leq C$$
$$\iff T^{\text{init}} \geq \left(\frac{\nu(B-1)}{C}\right)^{1/(1-\epsilon)}.$$

Thus, for a Matérn kernel, an initialization set D^{init} of size $T^{\text{init}} \geq \left(\frac{\nu(B-1)}{C}\right)^{1/(1-\epsilon)}$ implies that the conditional information gain of any future batch is $\leq C$. Choosing $C = 1$, we obtain the results presented in the corresponding row of Table 1.

B.3 Initialization Set Size: Squared-Exponential (RBF) Kernel

For the RBF kernel, the information gain is bounded by an expression similar to that of the linear kernel, $\gamma_t \leq \eta(\log(t+1))^{d+1}$ (Srinivas et al., 2010). Again, motivated by Inequality (15), one reasonable choice for an initialization set size is $T^{\text{init}} = k\eta(B-1)(\log B)^{d+1}$. We again attempt to show that there exists a finite k such that the conditional information gain of any post-initialization batch is $\leq C$. By a similar parallel argument to that for the linear kernel (Appendix B.1), and assuming that $B \geq 2$ and $k\eta(B-1) \geq 1$, it follows that

$$\frac{B-1}{T^{\text{init}}}\gamma_{T^{\text{init}}} \leq C \Longleftarrow \frac{\log k + \log\eta + \log(B-1)}{k^{1/(d+1)}(\log B)} \frac{\log\left[(\log B)^{d+1}+1\right]}{k^{1/(d+1)}(\log B)} \leq C^{1/(d+1)}$$
$$\Longleftarrow \frac{\log k}{k^{1/(d+1)}(\log B)} + \frac{\log\eta}{k^{1/(d+1)}(\log B)} + \frac{(d+2)}{k^{1/(d+1)}} \leq C^{1/(d+1)},$$

where the last implication follows because for $a \geq 0, b \geq 1, (a^b+1) \leq (a+1)^b$.

By noting that the maximum of $k^{-1/(d+1)}\log k$ over $k \in (0,\infty)$ is $(d+1)/e$ and choosing $C = (2(d+1)/e)^{d+1}$, we obtain for $k \geq 1/(\eta(B-1))$:

$$\frac{B-1}{T^{\text{init}}}\gamma_{T^{\text{init}}} \leq \left(\frac{2d+2}{e}\right)^{d+1} \Longleftarrow \frac{d+1}{e\log B} + \frac{1}{k^{1/(d+1)}}\left(\frac{\log\eta + (d+2)\log B}{\log B}\right) \leq \frac{2d+2}{e},$$

or equivalently, incorporating the constraint $k \geq 1/(\eta(B-1))$ explicitly,

$$\frac{B-1}{T^{\text{init}}}\gamma_{T^{\text{init}}} \leq \left(\frac{2d+2}{e}\right)^{d+1} \Longleftarrow k \geq \max\left[\frac{1}{\eta(B-1)}, \left(\frac{e(\log\eta + (d+2)\log B)}{(d+1)(2\log(B)-1)}\right)^{d+1}\right].$$

Thus, for a Squared-Exponential kernel and such a k, an initialization set D^{init} of size T^{init}, where $T^{\text{init}} \geq k\eta(B-1)(\log(B))^{d+1}$, ensures that the hallucinated conditional information in any future batch of size B is no more than $\left(\frac{2d+2}{e}\right)^{d+1}$.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2312–2320, 2011.

Jacob Abernethy, Elad Hazan, and Alexander Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, pages 263–274, 2008.

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

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

Javad Azimi, Alan Fern, and Xiaoli Fern. Batch Bayesian optimization via simulation matching. In *Advances in Neural Information Processing Systems (NIPS)*, pages 109–117, 2010.

Javad Azimi, Alan Fern, Xiaoli Fern, Glencora Borradale, and Brent Heeringa. Batch active learning via coordinated matching. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 1199–1206, 2012a.

Javad Azimi, Ali Jalali, and Xiaoli Fern. Hybrid batch Bayesian optimization. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 1215–1222, 2012b.

Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Computing Research Repository (CoRR)*, abs/1012.2599, 2010.

Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. Online optimization in X-armed bandits. In *Advances in Neural Information Processing Systems (NIPS)*, pages 201–208, 2008.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandit problems. In *Proceedings of the 20th International Conference on Algorithmic Learning Theory (ALT)*, pages 23–37, 2009.

Yuxin Chen and Andreas Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 160–168, 2013.

Dennis D. Cox and Susan John. SDO: A statistical method for global optimization. *Multidisciplinary Design Optimization: State of the Art*, pages 315–329, 1997.

Varsha Dani, Thomas P. Hayes, and Sham M. Kakade. Stochastic linear optimization under bandit feedback. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, pages 355–366, 2008.

Miroslav Dudik, Daniel Hsu, Satyen Kale, Nikos Karampatziakis, John Langford, Lev Reyzin, and Tong Zhang. Efficient optimal learning for contextual bandits. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 169–178, 2011.

Subhashis Ghosal and Anindya Roy. Posterior consistency of Gaussian process prior for nonparametric binary regression. *The Annals of Statistics*, 34(5):2413–2429, Oct. 2006.

David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. Kriging is well-suited to parallelize optimization. In Yoel Tenne and Chi-Keong Goh, editors, *Computational Intelligence in Expensive Optimization Problems*, volume 2 of *Adaptation, Learning, and Optimization*, pages 131–162. Springer Berlin Heidelberg, 2010.

John Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society, B*, 41(2):148–177, 1979.

Philipp Hennig and Christian J. Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research (JMLR)*, 13:1809–1837, June 2012.

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

Pooria Joulani, András György, and Csaba Szepesvári. Online learning under delayed feedback. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 1453–1461, 2013.

Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *ACM Symposium on Theory of Computing (STOC)*, pages 681–690. Association for Computing Machinery, Inc., May 2008.

Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *Machine Learning: ECML*, pages 282–293, 2006.

Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 324–331, 2005.

Andreas Krause and Cheng Soon Ong. Contextual Gaussian process bandit optimization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2447–2455, 2011.

Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research (JMLR)*, 9:235–284, February 2008.

Daniel Lizotte, Tao Wang, Michael Bowling, and Dale Schuurmans. Automatic gait optimization with Gaussian process regression. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 944–949, 2007.

Michel Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, pages 234–243. Springer, 1978.

Jonas Mockus. *Bayesian Approach to Global Optimization: Theory and Applications*. Kluwer Academic Publishers, 1989.

Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. *The application of Bayesian methods for seeking the extremum*, volume 2, pages 117–129. Elsevier, 1978.

Carl Edward Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research (JMLR)*, 11:3011–3015, 2010.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

Ilya O. Ryzhov, Warren B. Powell, and Peter I. Frazier. The knowledge gradient algorithm for a general class of online learning problems. *Operations Research*, 60(1):180–195, 2012.

Bernhard Schölkopf and Alex J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 2002.

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 1015–1022, 2010.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Information-theoretic regret bounds for Gaussian process optimization in the bandit setting. *Information Theory, IEEE Transactions on*, 58(5):3250–3265, 2012.

Christian Widmer, Nora C. Toussaint, Yasemin Altun, and Gunnar Rätsch. Inferring latent task structure for multitask learning by multiple kernel learning. *BMC Bioinformatics*, 11(Suppl 8):S5, 2010.