Optimistic Optimization of Gaussian Process Samples

Julia Grosse[1,*], Cheng Zhang[2], and Philipp Hennig[1,3]

[1]*University of Tübingen*
[2]*Microsoft Research Cambridge*
[3]*Max Planck Institute for Intelligent Systems*
[*]*Corresponding author: julia.grosse@uni-tuebingen.de*

September 5, 2022

Abstract

Bayesian optimization is a popular formalism for global optimization, but its computational costs limit it to expensive-to-evaluate functions. A competing, *computationally* more efficient, global optimization framework is *optimistic optimization*, which exploits prior knowledge about the geometry of the search space in form of a dissimilarity function. We investigate to which degree the conceptual advantages of Bayesian Optimization can be combined with the computational efficiency of optimistic optimization. By mapping the kernel to a dissimilarity, we obtain an optimistic optimization algorithm for the Bayesian Optimization setting with a run-time of up to $\mathcal{O}(N \log N)$. As a high-level take-away we find that, when using stationary kernels on objectives of relatively low evaluation cost, optimistic optimization can be strongly preferable over Bayesian optimization, while for strongly coupled and parametric models, good implementations of Bayesian optimization can perform much better, even at low evaluation cost. We argue that there is a new research domain between geometric and probabilistic search, i.e. methods that run drastically faster than traditional Bayesian optimization, while retaining some of the crucial functionality of Bayesian optimization.

1   Introduction

Bayesian optimization (BO) [Shahriari et al., 2015] is a popular and successful framework for global optimization. The foundation of most BO frameworks is a Gaussian Process (GP) regression method, whose kernel encodes prior knowledge about the objective function. This GP regressor provides a posterior over the objective, a probabilistic surrogate on which one can then reason about the extremum, and its location. This can be done in a variety of ways, e.g. by using the posterior to construct upper bounds (in GP-UCB Srinivas et al. [2009]), to estimate expected improvement (Močkus [1975]) or information gain (Hennig and Schuler [2012], Wang and Jegelka [2017]).

The computational cost of BO itself is significant, and arises from at least two sources: First, exact GP inference has cost $\mathcal{O}(N^3)$, where N is the number of observed function values ("samples"). Second, finding the next evaluation location requires a *continuous, numerical* optimization of the acquisition function. Since this utility function is generally multimodal, optimization should, at least in principle, be carried out on an increasingly fine grid, which contributes up to $\mathcal{O}(N^{2d})$ costs [Salgia et al., 2021] (a third source of cost is the construction of the acquisition function itself, but there some popular choices, like GP-UCB, for which this step is essentially a trivial sum of the mean and marginal standard-deviation constructed during inference, of negligible overhead). If the cost of individual function evaluations is very high, then the overhead of BO is irrelevant. But there are scenarios in which the cost of BO *is* a concern, namely when individual function evaluations are of intermediate cost, and (perhaps as a direct consequence), the total number N of evaluations is sufficiently large to make the cubic cost of GP inference noticeable. An example are settings involving computer simulations runs, e.g. in robotics, biology, chemistry and human-computer interaction design.

Considering this "middle ground" between sample and computational efficiency, we study a competing framework for global optimization, *optimistic optimization* (OO), which has drastically lower

computational overhead. OO does not require computing an explicit global posterior on the objective. However, OO can nevertheless leverage at least certain kinds of prior knowledge, captured in the form of a dissimilarity measure that can be constructed in the form of a probabilistic deviation inequality for samples from a GP. This in effect produces a map from a GP prior one might otherwise use in BO to a (much more time-efficient) OO model, so that prior knowledge encoded in the GP can be leveraged without the intermediate step of (cubically expensive) GP regression. For noiseless (or near-noiseless) observations, the resulting OO algorithm achieves $\mathcal{O}(N \log N)$ computational cost. Interestingly, *some* forms of prior information can not be leveraged in this way, so there *are* settings in which BO is indeed preferable, even in terms of raw overall speed.

In summary, we provide a practical map between BO and OO for global optimization (Section 3), resulting in a hybrid method we call GP-OO. We then contribute a formal analysis of this method in terms of regret (Section 5), also pointing out limitations in the process. Experiments (Section 6) corroborate that the proposed GP-OO can be significantly more time-efficient than BO in settings with low (but nevertheless realistic) function evaluation costs.

2   Background

2.1   Bayesian Optimization (BO)

We consider maximizing a function $f : \mathcal{X} \to \mathbb{R}$. Throughout, f^* denotes the maximum of the function, and x^*, the point where it is attained. $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a kernel function. The function f is assumed to be a sample from a GP $\mathcal{GP}(\mu, k)$. We assume that the GP is centered, i.e. $\mu = 0$. One has access to observations y_i, where $y_i \sim f(x_i) + \mathcal{N}(0, \lambda)$. The noiseless setting amounts to the assumption $\lambda \to 0$. After each new observation the posterior over the function is updated:

$$\mu_n(x) = \mathbf{k}_n(x)^T (\mathbf{K}_n + \lambda I)^{-1} \mathbf{y}_n \qquad k_n(x, x) = k(x, x) - \mathbf{k}_n(x)^T (\mathbf{K}_n + \lambda I)^{-1} \mathbf{k}_n(x), \qquad (1)$$

where $\mathbf{y}_n = [y_1, ..., y_n]^T, \mathbf{k}_n(x) = [k(x_1,), ..., k(x_n,)]^T$, and $\mathbf{K}_n$ is the Gram matrix with $K_{nij} = k_n(x_i, x_j)$. The posterior is used to define an acquisition function $a_n(x)$ at whose maximum the function is evaluated next. For an overview of different options of acquisition functions see Frazier [2018]. We will focus our analysis on the choice of GP-UCB Srinivas et al. [2009], where

$$a_n(x) = \mu_n(x) + \beta_n^{1/2} k_n(x, x) \quad \text{and} \quad \beta_n^{1/2} \text{ an appropriate constant.}$$

An advantage of GP-UCB over other choices is its comparatively simple structure, which facilitates not just implementation but also analysis. While other aquisition functions may well be more sample-efficient in concrete settings, this will not be important for what is to follow, because we are primarily concerned with run-time complexity, where GP-UCB is arguably indeed the fastest possible choice.

2.2   Optimistic Optimization (OO)

The optimistic optimization principle is used to optimize functions that are known to fulfill a local Lipschitz assumption with respect to a dissimilarity $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$:

$$\forall x \in \mathcal{X} : |f(x^*) - f(x)| \leq d(x^*, x). \qquad (2)$$

The method revolves around a hierarchical partitioning of the search space $\mathcal{X}$ that can be described by an infinite binary tree. Search can thus be implemented as a fast descent through the tree, thus only involves evaluations on a countable set of mesh points, in contrast to the local numerical optimization done within Bayesian optimization. This is one of the two reasons for the drastically shorter run-time costs of OO over BO (the other being that it does not require computing a GP posterior).

The root node corresponds to the entire search space $\mathcal{X}_{(0,1)} = \mathcal{X}$ and is named $(0, 1)$. Consider a node (t, i) at depth t. The left child $(t + 1, 2i - 1)$ and right child $(t + 1, 2i)$ represent two subregions $\mathcal{X}_{(t+1,2i-1)}$ and $\mathcal{X}_{(t+1,2i)}$ such that $\mathcal{X}_{(t,i)} = \mathcal{X}_{(t+1,2i-1)} \cup \mathcal{X}_{(t+1,2i)}$, i.e. the tree covers the entire space. To indicate that a cell was explored at iteration n, we refer to it as $\mathcal{X}_n$. Intuitively, it makes sense to select the cells in such a way that all points in a cell are similar to each other and all similar points are in the same cell. Formally, this can be expressed as

(a) $\forall x, y \in \mathcal{X}_{(t,i)} : \{d(x,y) < \delta(t)\}$

(b) $\exists x_{(t,i)} \in \mathcal{X}_{(t,i)} : \{y \in \mathcal{X} : d(x_{(t,i)}, y) < c\,\delta(t)\} \subset \mathcal{X}_{(t,i)}$,

where $\delta(t)$ is a decreasing sequence of diameters and c is a global constant. During search, the tree is build incrementally by adding the two children of a selected node. When a new node (t,i) is added, an observation is made at the center $x_{(t,i)}$ of the region $\mathcal{X}_{(t,i)}$. In each round, the leaf with the highest upper bound $U_{(t,i)} = f(x_{(t,i)}) + \delta(t)$ is selected for expansion. Since f is assumed to be locally Lipschitz with respect to d, selecting nodes by $U_{(t,i)}$ is a valid upper bound strategy (assuming noiseless observations). The sequence of $\delta(t)$ controls exploration. Therefore, the partitioning should be chosen in such a way, that the $\delta(t)$ can be as small as possible. The optimistic optimization principle is summarized in Algorithm 1 and Figure 1. Using a binary heap, the priority queue for the leaf nodes can be realized in $\mathcal{O}(N \log N)$ if the budget N is known in advance.

Algorithm 1 The optimistic optimization principle.

1: **procedure** OPTIMISTIC OPTIMIZATION(f, δ)
2:     priority-queue $\leftarrow [(0,1)]$ //root node
3:     **for** $j = 1$ to n **do**
4:         Select node (t, i) with maximum $U_{(t,i)} = f(x_{(t,i)}) + \delta(t)$ from the priority-queue
5:         Observe $f(x_{(t+1,2i-1)})$ and $f(x_{(t+1,2i)})$ for the two children of (t, i).
6:         Calculate child utilities $U_{(t+1,2i-1)} = f(x_{(t+1,2i-1)}) + \delta(t+1)$ and $U_{(t+1,2i)} = f(x_{(t+1,2i)}) + \delta(t+1)$.
7:         Add children $(t+1, 2i-1)$ and $(t+1, 2i)$ to the priority-queue based on their utilities.
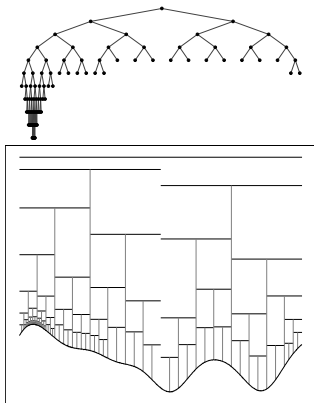8:   **end for**
9: **end procedure**

Figure 1: Optimistic optimization applied to a sample from a GP. The upper bounds are shown as horizontal bars. The vertical lines point to the evaluated locations.

This algorithm is a batch method in the sense that all children of a newly explored node are evaluated in each step. We assume batch size two, but other choices are possible too and might be preferable in some applications, e.g. when parallel evaluation is possible. A conceptual difference to the upper bound in GP-UCB is that one here works with upper bounds for entire *regions* of the search space, not for individual points in it. Also, the exploration term $\delta(t)$ in the upper bound is not updated based on new observations, but derived a priori. Nevertheless, the exploration term, aka. the uncertainty decreases during the search as the tree grows.

The hierarchical optimistic optimization principle has its origins in the Bandit setting. Bubeck et al. [2011] apply it in the noisy setting, Munos [2011] apply it in the noiseless setting and Kleinberg et al. [2008] uses it with a slightly different Lipschitz assumption. The assumptions on the dissimilarity d vary, e.g. some theoretical analyses require that it additionally is a semi-metric, or metric. Munos [2011] introduces a variant of the principle, simultaneous optimistic optimization (SOO), for situations in which the dissimilarity function is unknown. Valko et al. [2013] extend this idea to the noisy setting.

An interim summary: OO descends along a search tree, i.e. in a *discrete* sequence of steps, without numerical optimization, and only using *local* summary statistics, rather than updating a global posterior. This makes OO very fast, at least compared to Bayesian optimization. But the approach also has a downside: At least at first sight, it is not clear how to encode salient prior information about the global optimization problem into the algorithm. By contrast, many Bayesian optimization experts see the rich language of GP prior models as a key strength of their framework. The following section thus investigates formal connections between OO and BO. The goal is to understand to which degree the structural language of a GP prior can be translated into the algorithmic efficiency of OO.

3 Connection between Bayesian and Optimistic Optimization

The policy of OO is based on measuring similarity in the input domain in terms of a metric defined through function values. It turns out that Gaussian process models – or, more precisely, kernels – can be used immediately to define such a pseudo-metric (Section 3.1). We further show how the pseudo-metric can be used to obtain upper bounds on the supremum of the cell (Section 3.2), that allow for the application of the OO principle in the BO setting. Section 3.3 is concerned with how to choose the cells and in Section 3.4 we illustrate the derived concept (GP-OO) on concrete examples.

3.1 Canonical pseudo-metric

The canonical pseudo-metric $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for a centered GP is defined as

$$d(x,y) = \mathbb{E}_{f \sim \mathcal{GP}(0,k)}[(f(x) - f(y))^2] = \sqrt{k(x,x) + k(y,y) - 2k(x,y)}.$$

The fundamental relevancy of the canonical pseudo-metric arises from the fact that it controls the increments of the GP, in the sense of the following deviation inequality (Pisier [1999], Theorem 4.7):

$$\forall u > 0, \mathbb{P}(|f(x) - f(y)| \geq u) \leq 2 \exp\left(-\frac{u^2}{2d(x,y)^2}\right)$$

An important class of kernel-induced metrics is formed by monotonic transformations of the Euclidean metric, i.e. $d(x,y) = g(\|x - y\|_2)$ where $g : \mathbb{R} \mapsto \mathbb{R}$ is monotonically increasing. Many kernels used in practice are in this class, e.g. the square-exponential kernel, the Matérn class of kernels, the rational-quadratic kernels and the Wiener kernel as well as sums and products thereof. We refer to this class of kernels as $\mathcal{K}$. Kernels *violating* this assumptions are e.g. polynomial or periodic kernels.

3.2 Upper bound on the supremum of a cell

The first challenge in applying the optimistic optimization principle on samples of a GP consists in the probabilistic nature of the deviation inequality. To obtain a valid upper bound for the maximal deviation in a cell $\sup_{x \in \mathcal{X}_n} |f(x) - f(x_n)|$, the deviation for *all* points in the cell has to be bounded. We approximate such an upper bound by introducing two simplifications: We discretize the search space $\mathcal{X}$ into a finite number of points $\hat{\mathcal{X}}$ and we introduce an independence assumption between the $|f(x) - f(x_n)|$. Then we take a union bound approach:

$$\mathbb{P}(\sup_{x \in \hat{\mathcal{X}}_n} |f(x) - f(x_n)| \geq u) \qquad \leq \sum_{x \in \hat{\mathcal{X}}_n} \mathbb{P}(|f(x) - f(x_n)| \geq u) \qquad (3)$$

$$\leq 2 \sum_{x \in \hat{\mathcal{X}}_n} \exp\left(-\frac{u^2}{2d(x,y)^2}\right) \qquad \leq 2|\hat{\mathcal{X}}_n| \exp(-\frac{u^2}{2\Delta(\mathcal{X}_n)^2}) \qquad (4)$$

$$\text{where } \Delta(\mathcal{X}_n) = \max_{x \in \mathcal{X}_n} d(x_n, x). \qquad (5)$$

The bounds have to hold at each step n, so we additionally take a union bound over the number of steps. This implies the following statement, that holds with high probability :

$$\forall n : \sup_{x \in \hat{\mathcal{X}}_n} |f(x) - f(x_n)| \leq \beta_n^{1/2} \Delta(\mathcal{X}_n) \qquad (6)$$

where β_n are appropriate constants specified in the Supplements. The union bound approximation will be good if the $|f(x) - f(x_n)|$ are (nearly) uncorrelated, or the size of the discretization $|\hat{\mathcal{X}}|$ is chosen sufficiently small. Otherwise the bounds are loose, which leads to over-exploration. Though approximate, this idea of neglecting correlations to simplify the calculation of the expected supremum of dependent Gaussian variables is, for example, also done in Maximum Value Entropy Search [Wang and Jegelka, 2017] for BO or in related settings [Grosse et al., 2021]. The other extreme, a greedy approach with $\beta_n = 1$ has also been taken in recent work [Rando et al., 2022]. With generic chaining [Talagrand, 1996] it is possible to improve over the union bound approach. However, to the best of our knowledge state-of-the-art algorithms [Borst et al., 2020] to optimize for tighter bounds require polynomial run-time in the number of points per cell for arbitrary kernel functions. For special cases, like a Wiener kernel [Talagrand, 2021] or Matérn kernel functions [Shekhar and Javidi, 2018] analytical attempts to derive chaining based upper bounds exist.
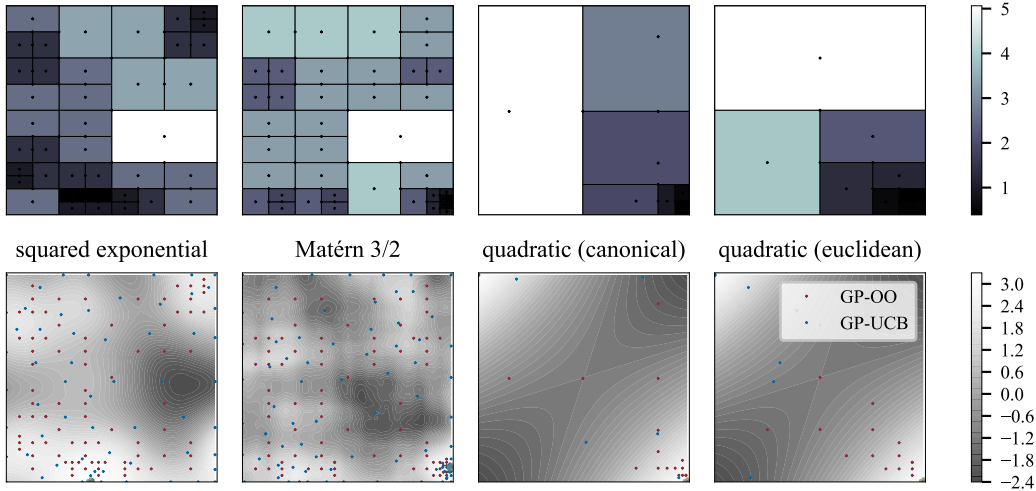
Figure 2: Top row: Cells and diameters Δ. Brighter colors indicate larger diameters. Bottom row: Sample from a GP and evaluation locations of GP-OO and GP-UCB.

3.3 Choosing the partitioning

The second main step in applying the OO principle is to choose the cells and location of the centers in such a way that the diameters of the cells are as small as possible. For k children nodes, this is a metric k-center problem – one of the classical NP-hard problems [Gonzalez, 1985]. A greedy approximation consists in iteratively picking the k centers with the largest distance to the previously picked centers, and requires $\mathcal{O}(|\hat{\mathcal{X}}|k)$ time. The greedy procedure is guaranteed to result in a 2-approximation, and there is no polynomial time algorithm doing better (unless P=NP). Working with a greedy instead of the optimal partitioning scheme thereby leads to an additional factor of 2 in the below regret bound, but is not harmful in the sense that the search gets stuck in a local optimum. NP-hardness also appears in the context of BO, e.g. the exploration term used in GP-UCB. And the acquisition functions of information-theoretic BO methods [Hennig and Schuler, 2012], [Wang and Jegelka, 2017] are related to the maximization of information gain, which is also a NP-hard problem.

From an implementation perspective, it is desirable to constrain the partitioning to axis-parallel boxes. For some kernel functions, e.g. the polynomial kernel, requirement (b) of the OO principle cannot be fulfilled with axis parallel boxes. One can nevertheless run the algorithm, but it will clearly be less information-efficient. One may even consider a randomized choice of centers, e.g. as done in Monte Carlo Tree Search [Chaslot et al., 2008]. However, it still remains to calculate the maximal distance from a point in the cell $\mathcal{X}$ to the center x_n. The computational complexity of this part is comparable to the numerical optimization of the acquisition function in BO. An advantage is that the domain over which one optimizes shrinks in each step. A disadvantage, though, is that if the numerical optimization is suboptimal and the maximal distance within a cell is underestimated, cells containing the optimum might get irreversibly pruned.

An important observation is that the partitioning scheme itself does not depend on the objective f, but only the kernel/distance function (how the search tree grows, however, *does* depend on f). This opens up the possibility of finding a good partitioning and the corresponding maximal distances analytically. For distances derived from kernels in $\mathcal{K}$, one can apply the following *regular* partitioning scheme: At each step, cut along the longest dimension in order to obtain the two children cells. Use the euclidean centers as centers. A point that maximizes the distance to the center will always be one of the 2^m corner points in m dimensions. Thus, for this type of kernels, the costs reduce to $\mathcal{O}(N2^m)$ for the partitioning, or $\mathcal{O}(N \log N + N2^m) = \mathcal{O}(N \log N)$ in total.

3.4 GP-OO

Motivated by the analysis above, we propose a variant of OO, which we call GP-OO. It consists in running Algorithm 1 with the utility $\mathcal{U}_{(n)} = f(x_{(n)}) + \beta_n^{1/2}\Delta(\mathcal{X}_n)$ in line 4, where Δ is as defined in Eq. (5). For kernels that match the observation above, one can then still choose the evaluation node

(in $\mathcal{O}(2^m)$ time) without numerical optimization. Figure 2 shows the algorithm running on GP samples from a square-exponential and a Matérn kernel with $\nu = 2/3$ on the domain $[0,5]^2$, as well as from a quadratic kernel on the domain $[-1,1]^2$. In regions with higher function values the partitions are finer. The Matérn kernel yields higher distances than the square exponential, leading to more exploration, reflecting that the samples are less smooth. By optimizing the partitions and centers of the cells with respect to the canonical pseudo-metric, larger parts of the search space can be covered while keeping the cell diameters constant as shown by the two examples with the quadratic kernel.

4   Related Work

In this section we review related work at the intersection of OO and BO. With the exception of Grill et al. [2018], the fundamental difference to all of this work is that we do not keep track of a GP-posterior, thus saving significant computational cost (see Figure 3).

Work without the canonical pseudo-metric. BO methods have been combined with SOO (Munos [2011]), the version of OO with unknown dissimilarity. In BamSOO, Wang et al. [2014] use GP-UCB to reduce the number of evaluations required when running SOO alone. By using SOO, they can in return reduce the optimization costs of the acquisition function. Gupta et al. [2021] improve upon the basic version of BamSOO by a more elaborate partitioning scheme: Instead of dividing a cell into k children along the longest side of the cell, they divide along the b longest dimensions into a cells, where $b^a = k$. Salgia et al. [2021] use a random walk based strategy on a tree to improve over the grid-based optimization of the GP-UCB acquisition function. For the Matérn and Squared Exponential kernel, they achieve order optimal regret, but the computational complexity is $\mathcal{O}(N^4)$.

Work with the canonical pseudo-metric. Shekhar and Javidi [2018] use the GP's canonical pseudo-metric to improve the numerical optimization by pruning the search regions. They additionally keep track of the posterior to only evaluate at locations where posterior uncertainty exceeds the cell's upper bound. Rando et al. [2022] follow this approach and additionally introduce a Nyström approximation, which allows for approximate inference in $\mathcal{O}(N^2 d_{\text{eff}}^2)$, where d_{eff} is the effective dimension of the search space. Contal et al. [2015] replace the GP-UCB bounds with bounds derived from the pseudo-metric, but do not use a hierarchical approach, i.e. they construct bounds for individual points. They update the posterior and the exploration terms after every new observation.

Grill et al. [2018] apply the optimistic optimization principle to a one-dimensional Brownian walk. A minor difference is, that they evaluate a cell at the corners of an interval and not in the center. There are cases, where this is advantageous, e.g. think of samples from a GP with a polynomial or linear kernel. However, the number of corners scales exponentially with the dimension.

5   Regret

While computational and not sample efficiency is our main motivation to apply the OO principle in the BO setting, we show that the resulting method nevertheless leads to non-trivial regret. In particular, it is asymptotically regret-free in the limit $\lim_{N \to \infty} R_N / N$. Here, R_N denotes the cumulative regret defined as $R_N = \sum_{n=0}^{N} f(x^*) - f(x_n)$.

Building upon arguments from Munos [2011] and Shekhar and Javidi [2018], we obtain the following guarantee for the cumulative regret:

Proposition 1 *Let $\mathcal{X}$ be finite, $\epsilon \in (0,1)$ and $\beta_n = 2 \log(N|\mathcal{X}_n|/2\epsilon)$. Running GP-OO with β_n for a sample f from a GP with mean function zero and covariance function $k(x,x)$, the following regret bound holds with probability $1 - \epsilon$:*

$$R_N \leq \beta^{1/2} \sum_{n=1}^{N} \Delta(\log n)$$

where $\beta = max\{\beta_1, ..., \beta_N\}$ and $\Delta(n)$ denotes the diameter of a cell evaluated at depth n.

A full proof is provided in the Supplements. The high-level idea is that either the explored cell $\mathcal{X}_n$ contains the optimum x^*, then the simple regret $|f(x^*) - f(x)|$ is trivially upper bounded by the maximal deviation $\beta^{1/2} \Delta(\mathcal{X}_n)$. Or the cell does not contain the optimum, but then then its utility was higher than the one of a node containing the optimum in its region, and thereby higher than the
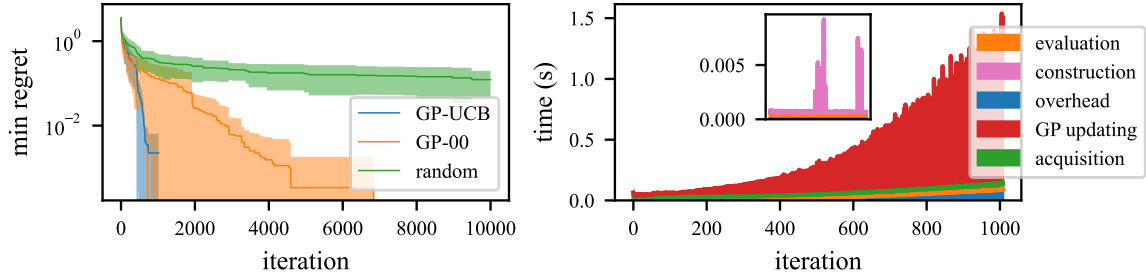
Figure 3: Left: Minimum simple regret $|f(x^*) - f(x)|$ on synthetic samples. Right: Composition of the average computational costs in GP-UCB (large panel) and GP-OO (small panel, note that its ordinate is re-scaled by a factor of 10^3).

optimum itself. For the cumulative regret we assume the worst-case of a uniformly growing tree. For a broad class of kernels, the bound in Proposition 1 can be further specified:

Proposition 2 *Assume the GP's canonical pseudo-metric d satisfies $d(x, y) = C\|x - y\|_2^\alpha$, where $C > 0, \alpha > 0, m/\alpha > 1$. Running GP-OO on a finite domain $\mathcal{X} \subset [0,1]^m$ with regular partitions, one has a worst-case cumulative regret R_N of*

$$\tilde{\mathcal{O}}(N^{1-\alpha/m}(\log N)^{\alpha/m})$$

with high probability The $\tilde{\mathcal{O}}(\cdot)$ notation supresses poly-logarithmic factors.

In particular, for the squared exponential kernel and Matérn kernels with half-integer values $\nu \geq 3/2$, one has $\alpha = 1$ and $R_N \in \tilde{\mathcal{O}}(N^{1-1/m}(\log N)^{1/m})$. For comparison, the regret in GP-UCB grows as $\tilde{\mathcal{O}}(\sqrt{N}\log(N)^{\frac{m+1}{2}})$ for the squared exponential kernels and thereby scales better to higher dimensions. For the Matérn class, GP-UCB is guaranteed to have regret at most $\tilde{\mathcal{O}}(N^{\frac{\nu+m(m+1)}{2\nu+m(m+1)}})$ for $\nu \geq 1$. Our bound is tighter, e.g., for the values $\nu = 3/2$ or $\nu = 5/2$ often used in practice.

Shekhar and Javidi [2018] establish regret bounds in terms of the near-optimality dimension $\tilde{m}$. This measure is commonly used in optimistic optimization to characterize the size of the set of ϵ-optimal points in terms of packing numbers. The near-optimality dimension does not only depend on the underlying metric, but also on the function f itself and is thereby a random variable. Smaller values are associated with deeper growing trees, whereas larger values lead to more balanced, uniformly growing trees. Assuming the worst case of $\tilde{m} = m$, the bounds in Shekhar and Javidi [2018] become $\tilde{\mathcal{O}}(N^{1-\alpha/m})$ for the Matérn class and match our worst-case bound. However, we restricted the analysis to finite domains $\mathcal{X}$, and $|\mathcal{X}|$ enters our bound logarithmically.

Grill et al. [2018] showed that in the specific case of Brownian motion, the tree built during optimistic optimization does not grow with the worst-case uniform rate.

6 Experiments

We empirically compare GP-OO to GP-UCB in terms of regret and time, on synthetic three dimensional samples from Gaussian processes, and Benchmark functions from Surjanovic and Bingham [2022]. All experiments were performed on a desktop machine (hardware details in the Supplements). Since computational efficiency is the key concern of our analysis, we often report performance relative to wall-clock time. We recognise, however, that such results are implementation dependent, and should be interpreted qualitatively. For GP-UCB, we use implementations from Emukit [Paleyes et al., 2019, Gardner et al., 2018].

6.1 Experiments on synthetic functions

Regular partitions. We begin with on-model experiments for a common setting in Bayesian optimization, where GP-OO can showcase its speed advantages: 20 samples from a GP with squared exponential kernel (lengthscale $l = 0.1$) on the unit cube $\mathcal{X} = [0,1]^3$. For all experiments, the noise level for GP-UCB is set to a very small constant $\lambda = 0.001$ since we assume noiseless observations.
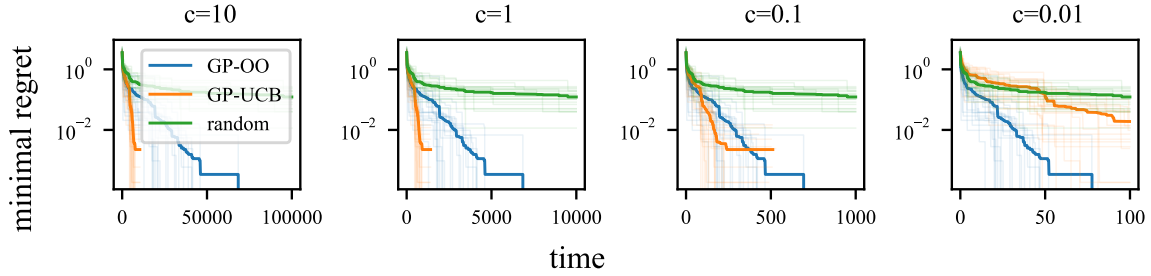
Figure 4: Minimum simple regret $|f(*) - f(x)|$ on synthetic samples, for different function evaluation costs c, from high (10sec, left) to low (0.01sec, right). GP-UCB becomes less competitive as evaluation cost drops. For extremely cheap functions, purely random search eventually outperforms everything, because it can simply cover the whole region at nearly no overhead. Note the varying abscissas between plots. The line for GP-UCB changes nontrivially, reflecting its significant numerical overhead compared to the other two methods. Details in text.

For GP-OO, we use $\beta_n = 2\log(|\hat{\mathcal{X}}_n| N / 2\epsilon|)$ with fixed horizon $N = 10000$, and for GP-UCB, we use $\beta_n = 2\log(|\hat{\mathcal{X}}| n^2 \pi^2 / 6\epsilon)$ according to theory. We did a grid search over the size of the discretization $|\hat{\mathcal{X}}|$ with values in $\{1, 10, 100, 1000\}$. The confidence level ϵ is set to 0.01 for both methods, and we set $|\hat{\mathcal{X}}| = 1$. Even though GP-UCB is more sample-efficient than GP-OO (Figure 3, left) GP-OO is orders of magnitude faster per step (Figure 3, right).

This suggests there is a "sweet-spot": We artificially added $c \in \{0.01, 0.1, 1, 10\}$ seconds to the per-evaluation costs to simulate different evaluation costs (Figure 4). For evaluation time around 0.1 seconds and below, GP-OO becomes faster than GP-UCB. When the objective becomes "expensive", BO can leverage its sample efficiency (at the very extreme end of nearly instant function-evaluations, random search, which has negligible overhead, is always a trivially optimal asymptotic baseline).

Non-regular partitions. For demonstration, we also explore a setting with non-regular optimal partitions, where GP-OO can not perform as well. We sample 100 functions from a GP with quadratic kernel $k(x, y) = (x^T y)^2$ with bias 0 on the domain $[-1, 1]^3$. We consider GP-UCB and GP-OO, once with euclidean partitions and once with partitions optimized with respect to the canonical metric. The exploration constant was set to $\beta = 1$. The results shown (Figure 5) show that it is advantageous to optimize the partitioning scheme with respect to the canonical metric. However, we restrict ourselves to partitions with axis-aligned cells, i.e. the perfectly correlated corner points with $k(x, y) = 1$ and $d(x, y) = 0$ end up in different cells. At this point, GP-UCB has a clear advantage, as information can be shared across the search space between the corner points. Also, the run-time advantage decreases due to the numerical optimization of the partitions.
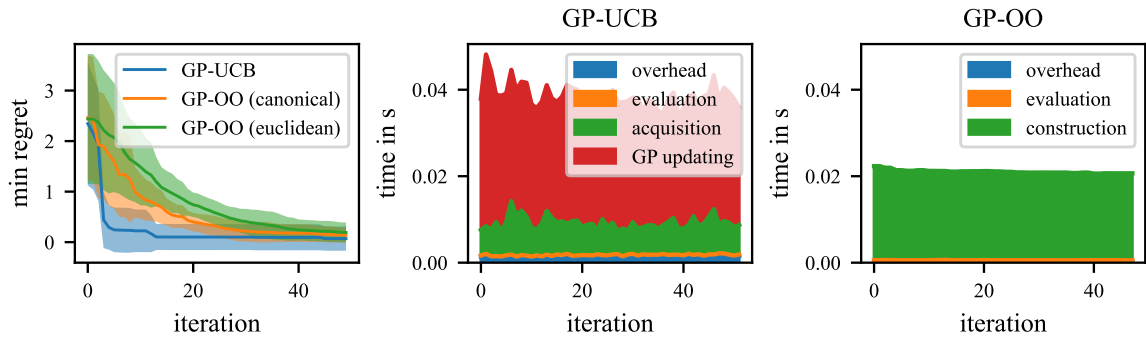

Figure 5: Experiment with a quadratic kernel, a problem specifically chosen such that GP-OO can not be expected to work well. Left: Performance in terms of the minimal obtained simple regret $|f(x^*) - f(x)|$. Middle: Wall-clock time with GP-UCB. Right: Wall-clock time with GP-OO.
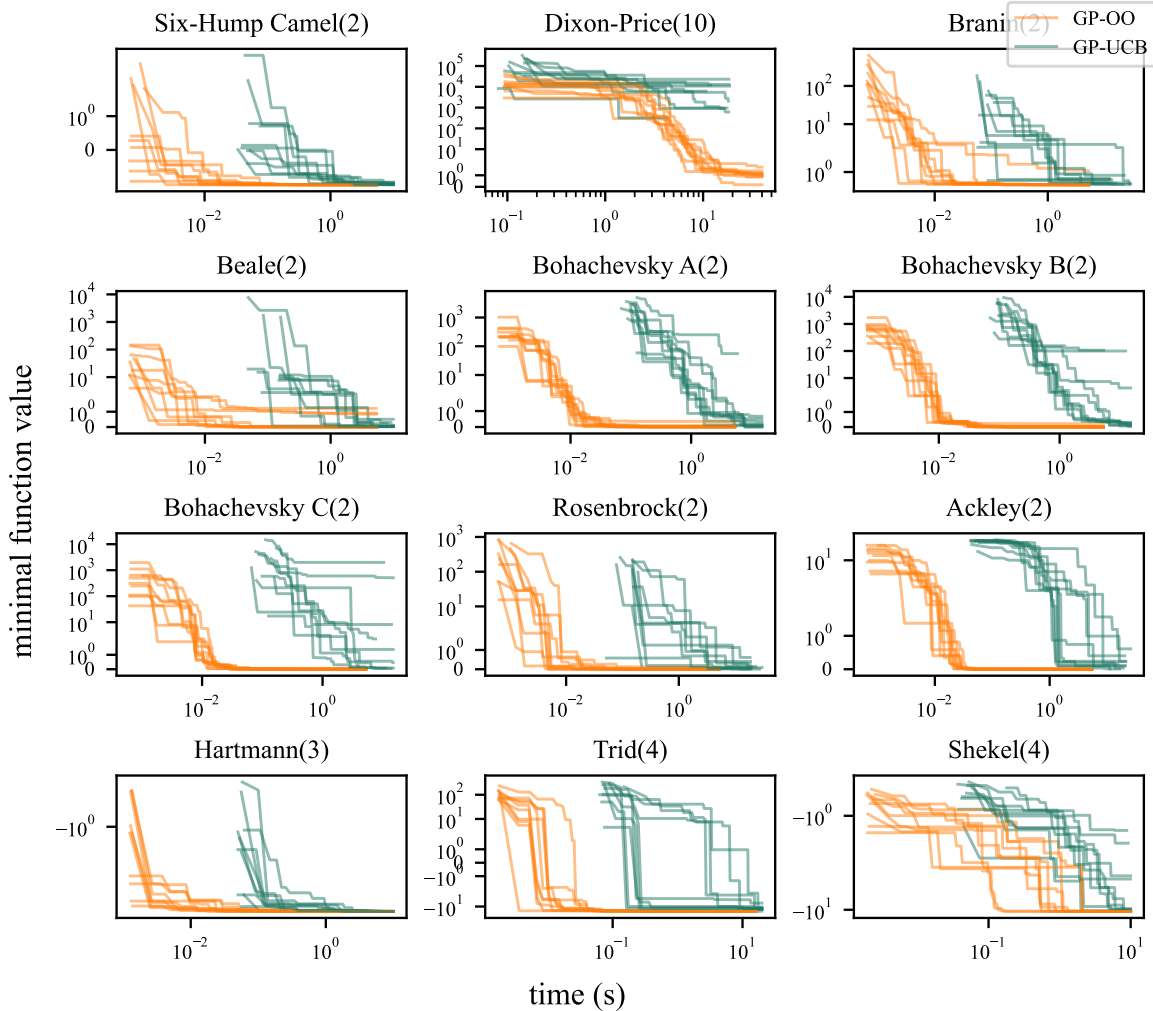
8

Figure 6: Optimization performance (minimal function values found) on common benchmarks with GP-UCB and GP-OO. The number in brackets indicates the dimension of the search domain. Note that all plots show wall-clock time in log-scale, not number of function evaluations, highlighting the strong *computational* advantage of GP-OO over GP-UCB (as opposed to sample efficiency).

6.2 Experiments on benchmarks

We consider the minimization of common optimization benchmark functions, on domains with dimensions ranging from 2 to 10. Since the functions are deterministic, and GP-OO is deterministic up to random tie breaks, we randomly sample 10 domains $\mathcal{X}$ to obtain variation (see Supplements). We use a Matérn kernel with $\nu = 3/2$ and length-scales as recommended by Rando et al. [2022]. For the exploration constant β we did a grid search over $\{0.1, 1, 10, 100\}$. We use confidence level $\epsilon = 0.05$. Figure 6 shows minimal attained function value over time. Here, GP-OO outperforms GP-UCB in many cases. This is not just due to runtime, but in fact also sometimes holds in function evaluations (see Supplements). Here, both methods did not always find the global optimum (which can not be guaranteed here as these functions are not necessarily within the RKHS).

7 Conclusion

BO and OO are closely connected through a mapping from the kernel-function to the canonical pseudo-metric of a GP. We showed that, for some, though certainly not all kernels, this connection can be exploited to derive a computationally efficient method for the BO setting that captures prior

information. For common choices of kernels, like the Matérn class, we outperform even computationally light-weight BO approaches, like GP-UCB, if the objective function has reasonably low evaluation cost. Where the objective function is truly *very* expensive, BO remains competitive. And, for strongly coupled models, too, we find BO to be advantageous. Our work shows that the ability to use prior information in Bayesian optimization does not *have* to be expensive per se, but can also be achieved without explicitly tracking a posterior.

8 Acknowledgements

References

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

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

Jonas Močkus. On Bayesian methods for seeking the extremum. In *Optimization techniques IFIP technical conference*, pages 400–404. Springer, 1975.

Philipp Hennig and Christian J Schuler. Entropy Search for Information-Efficient Global Optimization. *Journal of Machine Learning Research*, 13(6), 2012.

Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient Bayesian optimization. In *International Conference on Machine Learning*, pages 3627–3635. PMLR, 2017.

Sudeep Salgia, Sattar Vakili, and Qing Zhao. A Domain-Shrinking based Bayesian Optimization Algorithm with Order-Optimal Regret Performance. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

Peter I Frazier. A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. X-armed Bandits. *Journal of Machine Learning Research*, 12(5), 2011.

Rémi Munos. Optimistic optimization of a deterministic function without the knowledge of its smoothness. *Advances in neural information processing systems*, 24:783–791, 2011.

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

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

Gilles Pisier. *The volume of convex bodies and Banach space geometry*, volume 94. Cambridge University Press, 1999.

Julia Grosse, Cheng Zhang, and Philipp Hennig. Probabilistic DAG search. In *Uncertainty in Artificial Intelligence*, pages 1424–1433. PMLR, 2021.

Marco Rando, Luigi Carratino, Silvia Villa, and Lorenzo Rosasco. Ada-BKB: Scalable Gaussian Process Optimization on Continuous Domains by Adaptive Discretization. In *International Conference on Artificial Intelligence and Statistics*, pages 7320–7348. PMLR, 2022.

Michel Talagrand. Majorizing measures: the generic chaining. *The Annals of Probability*, 24(3): 1049–1103, 1996.

Sander Borst, Daniel Dadush, Neil Olver, and Makrand Sinha. Majorizing measures for the optimizer. *arXiv preprint arXiv:2012.13306*, 2020.

Michel Talagrand. Empirical processes, ii. In *Upper and Lower Bounds for Stochastic Processes*, pages 433–456. Springer, 2021.

Shubhanshu Shekhar and Tara Javidi. Gaussian process bandits with adaptive discretization. *Electronic Journal of Statistics*, 12(2):3829–3874, 2018.

Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.

Guillaume Chaslot, Sander Bakkes, Istvan Szita, and Pieter Spronck. Monte-Carlo tree search: A new framework for game ai. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 4, pages 216–217, 2008.

Jean-Bastien Grill, Michal Valko, and Rémi Munos. Optimistic optimization of a Brownian. *Advances in Neural Information Processing Systems*, 31, 2018.

Ziyu Wang, Babak Shakibi, Lin Jin, and Nando Freitas. Bayesian multi-scale optimistic optimization. In *Artificial Intelligence and Statistics*, pages 1005–1014. PMLR, 2014.

Sunil Gupta, Santu Rana, Svetha Venkatesh, et al. Bayesian Optimistic Optimisation with Exponentially Decaying Regret. In *International Conference on Machine Learning*, pages 10390–10400. PMLR, 2021.

Emile Contal, Cédric Malherbe, and Nicolas Vayatis. Optimization for Gaussian processes via chaining. *arXiv preprint arXiv:1510.05576*, 2015.

S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets. Retrieved May 16, 2022, from `http://www.sfu.ca/~ssurjano`, 2022.

Andrei Paleyes, Mark Pullin, Maren Mahsereci, Neil Lawrence, and Javier González. Emulation of physical processes with emukit. In *Second Workshop on Machine Learning and the Physical Sciences, NeurIPS*, 2019.

Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018:7576–7586, 2018.

8.1 Theoretical Analysis

8.1.1 Background

The following three facts will be used during the analysis:

1. **Deviation inequality** For a sample $f \sim \mathcal{G}(0, k)$ from a centered GP with kernel function k, it holds

$$\forall u, \mathbb{P}(|f(x) - f(y)| \geq u) \leq 2 \exp\left(-\frac{u^2}{2d(x,y)^2}\right), \tag{7}$$

   where $d(x, y) = \sqrt{k(x, x) + k(y, y) - 2k(x, y)}$.

2. **Hölder's inequality** Let $a_1, ..., a_N, b_1, ..., b_N$ be real numbers.

$$\sum_{n=1}^{N} |a_n b_n| \leq \left(\sum_{n=1}^{N} |a_n|^p\right)^{1/p} \left(\sum_{n=1}^{N} |b_k|^q\right)^{1/q} \tag{8}$$

   with $p, q \geq 1$ and $\frac{1}{p} + \frac{1}{q} = 1$

3. **Growth of Harmonic numbers** The N-th Harmonic number $H_N = \sum_{n=1}^{N} \frac{1}{n}$ grows logarithmically in N:

$$H_N \in \Theta(\log N) \tag{9}$$

8.1.2 Upper bounds on the supremum of a cell

Lemma 1. Assume $\mathcal{X}$ is finite and $f \sim \mathcal{GP}(0, k)$. Pick $\epsilon \in (0, 1)$ and set $\beta_n = 2\log(2|\mathcal{X}_n|N/\epsilon)$, where $\sum_{n \geq 1} \pi_n^{-1} = 1, \pi_n > 0$. Then

$$\forall n \sup_{x \in \mathcal{X}_n} |f(x) - f(x_n)| \leq \beta_n^{1/2} \Delta(\mathcal{X}_n)$$

holds with probability $\geq 1 - \epsilon$. $\mathcal{X}_n$ is the cell visited at step n with center x_n and $\Delta(\mathcal{X}_n) = \sup_{x \in \mathcal{X}_n} d(x, x_n)$.

Proof of Lemma 1. Fix n. Applying a union bound and 7, one obtains for all u_n

$$\mathbb{P}(sup_{x \in \mathcal{X}_n} |f(x) - f(x_n)| \geq u_n) \tag{10}$$

$$\leq \sum_{x \in \mathcal{X}_n} \mathbb{P}(|f(x) - f(x_n)| \geq u_n) \tag{11}$$

$$\leq 2 \sum_{x \in \mathcal{X}_n} \exp\left(-\frac{u_n^2}{2d(x, x_n)^2}\right) \tag{12}$$

$$\leq 2|\mathcal{X}_n| \exp\left(-\frac{u_n^2}{2\Delta(\mathcal{X}_n)^2}\right) \tag{13}$$

For $u_n = \beta_n^{1/2} \Delta(\mathcal{X}_n)$, it holds with probability $1 - \epsilon/N$, that $sup_{x \in \mathcal{X}_n} |f(x) - f(x_n)| \leq \beta_n^{1/2} \Delta(\mathcal{X}_n)$. Taking another union bound over N the statement holds.

8.1.3 Upper bound on the regret

Lemma 2. Running GP-OO with β_n as specified in Lemma 1 and the canonical pseudo-metric d, the simple regret $r_n = f^* - f_n$ is bounded by $\beta_n^{1/2} \Delta(\mathcal{X}_n)$ for all n with probability $1 - \epsilon$.

Proof of Lemma 2. This statement can be shown with typical arguments from the literature on optimistic optimization. We consider the cases $x^* \in \mathcal{X}_n$ and $x^* \notin \mathcal{X}_n$ separately.

Case 1: $x^* \in \mathcal{X}_n$

$$r_n = f^* - f_n = sup_{x \in \mathcal{X}_n} |f(x) - f(x_n)| \leq \beta_n^{1/n} \Delta(\mathcal{X}_n) \tag{14}$$

by Lemma 1.

Case 2: $x^* \notin \mathcal{X}_n$.
Because x_n was explored nevertheless, there is a node $x_{n'}$ on the optimal path with $x^* \in \mathcal{X}_{n'}$, that was explored at step $n' < n$, such that

$$f(x_n) + \beta_n^{1/2}\Delta(\mathcal{X}_n) \geq f(x_{n'}) + \beta_{n'}^{1/2}\Delta(\mathcal{X}_{n'}) \tag{15}$$

Then, $f(x_{n'}) - f(x_n) \leq \beta_n^{1/2}\Delta(\mathcal{X}_n) - \beta_{n'}^{1/2}\Delta(\mathcal{X}_{n'})$. For the regret one obtains in combination with Lemma 1:

$$r_n = f^* - f_n \tag{16}$$
$$\leq \left[f(x^*) - f(x_{n'})\right] + \left[f(x_{n'}) - f(x_n)\right] \tag{17}$$
$$\leq \left[\beta_{n'}^{1/2}\Delta(\mathcal{X}'_n)\right] + \left[\beta_n^{1/2}\Delta(\mathcal{X}_n) - \beta_{n'}^{1/2}\Delta(\mathcal{X}_{n'})\right] \tag{18}$$
$$= \beta_n^{1/2}\Delta(\mathcal{X}_n) \tag{19}$$

Proposition 1. Assume $\mathcal{X}$ is finite and $f \sim \mathcal{GP}(0, k)$. Pick $\epsilon \in (0, 1)$ and set $\beta_n = 2\log(2|\mathcal{X}_n|N/\epsilon)$. For GP-OO with $k = 2$ children nodes, we obtain the following bound on the cumulative regret

$$\mathbb{P}(R_N \leq \beta^{1/2}\sum_{n=1}^{N}\Delta(\lfloor\log n\rfloor)) \geq 1 - \epsilon \tag{20}$$

where $\Delta(h)$ is the radius of a cell at depth h and $\beta = \max\{\beta_1, ..., \beta_N\}$.

Proof Proposition 1. Simple consequence from Lemma 2, where we assume the worst case of a uniformly growing tree. The depth of a node after n steps is at least $\lfloor\log(n)\rfloor$ in a uniformly growing tree.

8.1.4 Bounds for common kernels

The following analysis is restricted to GP's, where the canonical pseudo-metric d satisfies:

Assumption 1. There exist $C > 0, \alpha > 0$ such that $d(x, y) \leq C\|x - y\|_2^\alpha$, where $\|\cdot\|_2$ is the Euclidean norm. We additionally require $m/\alpha > 1$, where m is the dimension of the domain.

According to Shekhar and Javidi [2018] the first part of Assumption 1 holds for the squared exponential kernel with $C = \sqrt{2/l}, \alpha = 1$ and the Matern kernels with half integer values. For $\nu = 1/2$, one has $\alpha = 1/2$ and for all other half-integer values $\alpha = 1$.

Lemma 3. [Bubeck et al. [2011]] Assume that $\mathcal{X}$ is a m-dimensional hypercube $[0, 1]^m$ and consider the dissimilarity $d(x, y) = C\|x - y\|_2^\alpha$, where $C > 0, \alpha > 0$. Define the partitioning by recursively splitting the hypercube in the middle along its longest side (ties broken arbitrarily). One has

$$\Delta(\mathcal{X}_n) \leq diam(\mathcal{X}_h) \leq C(2\sqrt{m})^\alpha\left(\frac{1}{2^{\alpha/m}}\right)^h$$

for the cell $\mathcal{X}_h$ of a node at depth h.

Proof of Lemma 3. See Example 1 in Bubeck et al. [2011].

Proposition 2 *Assume the GP's canonical pseudo-metric d satisfies $d(x, y) = C\|x - y\|_2^\alpha$, where $C > 0, \alpha > 0, m/\alpha > 1$. Running GP-OO on a finite domain $\mathcal{X} \subset [0, 1]^m$ with regular partitions, one has a worst-case cumulative regret R_N of*

$$\tilde{\mathcal{O}}(N^{1-\alpha/m}(\log N)^{\alpha/m})$$

with high probability. The $\tilde{\mathcal{O}}(\cdot)$ notation supresses poly-logarithmic factors.

Proof of Proposition 2 It follows from Proposition 1 that $R_N \in \tilde{\mathcal{O}}(\sum_{n=1}^{N}\Delta(\lfloor\log n\rfloor))$ with probability $1 - \epsilon$. It remains to bound $\sum_{n=1}^{N}\Delta(\lfloor\log n\rfloor)$. For Equation (21), we use Lemma 3 and for

Equation (26), we apply Hölder's inequality (8) with $q = m/\alpha$ and $p = \frac{1}{1-\alpha/m}$.

$$\sum_{n=1}^{N} \Delta(\lfloor \log n \rfloor) \leq C\left(2\sqrt{m}\right)^{\alpha} \sum_{n=1}^{N} \left(\frac{1}{2^{\alpha/m}}\right)^{\lfloor \log n \rfloor} \tag{21}$$

$$\leq C(2\sqrt{m})^{\alpha} \sum_{n=1}^{N} \left(\frac{1}{2^{\alpha/m}}\right)^{\log n - 1} \tag{22}$$

$$= C(2\sqrt{m})^{\alpha} 2^{\alpha/m} \sum_{n=1}^{N} \left(\frac{1}{2^{\alpha/m}}\right)^{\log n} \tag{23}$$

$$= C_1 \sum_{n=1}^{N} \left(\frac{1}{2^{\alpha/m}}\right)^{\log n} \tag{24}$$

$$= C_1 \sum_{n=1}^{N} \left(\frac{1}{n^{\alpha/m}}\right) \tag{25}$$

$$\leq C_1 \left(\sum_{n=1}^{N} 1\right)^{1-\alpha/m} \left(\sum_{n=1}^{N} \left(\frac{1}{n^{\alpha/m}}\right)^{m/\alpha}\right)^{\alpha/m} \tag{26}$$

$$= C_1 N^{1-\alpha/m} H_N^{\alpha/m} \tag{27}$$

where $C_1 = C(2\sqrt{m})^{\alpha} 2^{\alpha/m}$ and H_N being the N-th harmonic number. Together with (9), this implies:

$$\sum_{n=1}^{N} \Delta(\lfloor \log n \rfloor) \in \mathcal{O}(N^{1-\alpha/m} (\log N)^{\alpha/m}) \tag{28}$$

8.2 Additional experimental details

All experiments were implemented in Python 3.9.1. and run on a machine with macOS 12.3.1, a 4 GHz Quad-Core Intel Core i7 CPU and 32 GB RAM.

8.2.1 Experiment with benchmark functions

Table 1 lists the domains and hyperparameters used in the experiment with the benchmark functions. For each run, we sampled a sub-domain by choosing uniformly at random new lower and upper boundaries for the intervals along each dimension, such that the new boundaries are inbetween the previous ones and the location of the minimum. In this way, the location of the minimum stays the same as in the original domain. Figure 7 shows the minimal function value found for the benchmark functions.

Table 1: Hyperparameters for experiment with benchmark functions

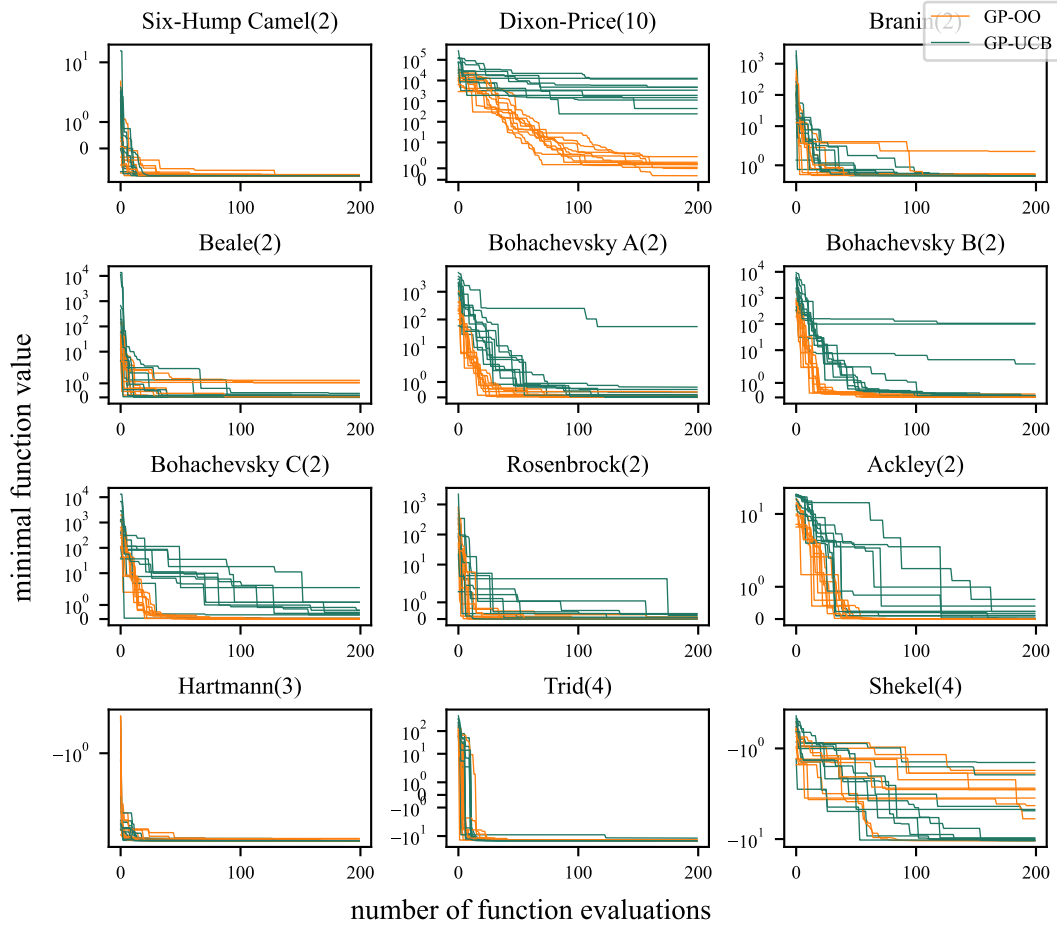| Benchmark	Domain	Lengthscale l	β (GP-UCB)	β (GP-OO)
Branin	$[-15, 15]^2$	0.5	10	100
Six-Hump-Camel	$[-2, 2]^2$	0.5	1	100
Beale	$[-4.5, 4.5]^2$	1	1	100
Bohachevsky a	$[-35.5, 100]^2$	1.7	10	10
Bohachevsky b	$[-35.5, 100]^2$	1.7	10	10
Bohachevsky c	$[-35.5, 100]^2$	1.7	100	10
Rosenbrock	$[-3, 3]^2$	0.7	1	100
Ackley	$[-12.5, 35]^2$	3.5	10	10
Hartmann	$[0, 1]^3$	0.3	1	10
Trid	$[-16, 16]^4$	10.75	0.1	100
Shekel	$[0, 10]^4$	1.75	10	10
Dixonprice	$[-10, 10]^{10}$	2	1	10

Figure 7: Minimal function values found on common benchmark datasets.