High Dimensional Bayesian Optimisation and Bandits via Additive Models

Kirthevasan Kandasamy KANDASAMY@CS.CMU.EDU
Jeff Schneider SCHNEIDE@CS.CMU.EDU
Barnabás Póczos BAPOCZOS@CS.CMU.EDU
Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

Bayesian Optimisation (BO) is a technique used in optimising a D-dimensional function which is typically expensive to evaluate. While there have been many successes for BO in low dimensions, scaling it to high dimensions has been notoriously difficult. Existing literature on the topic are under very restrictive settings. In this paper, we identify two key challenges in this endeavour. We tackle these challenges by assuming an additive structure for the function. This setting is substantially more expressive and contains a richer class of functions than previous work. We prove that, for additive functions the regret has only linear dependence on D even though the function depends on all D dimensions. We also demonstrate several other statistical and computational benefits in our framework. Via synthetic examples, a scientific simulation and a face detection problem we demonstrate that our method outperforms naive BO on additive functions and on several examples where the function is not additive.

1. Introduction

In many applications we are tasked with zeroth order optimisation of an expensive to evaluate function f in D dimensions. Some examples are hyper parameter tuning in expensive machine learning algorithms, experiment design, optimising control strategies in complex systems, and scientific simulation based studies. In such applications, f is a blackbox which we can interact with only by querying for the value at a specific point. Related to optimisation is the bandits problem arising in applications such as online advertising and reinforcement learning. Here the objective is to maximise the cumulative sum of all queries. In either case, we need to find the optimum of f using as few queries as possible by managing exploration and exploitation.

Bayesian Optimisation (Mockus & Mockus, 1991) refers to a suite of methods that tackle this problem by modeling f as a Gaussian Process (GP). In such methods, the challenge is two fold. At time step t, first estimate the unknown f from the query value-pairs. Then use it to intelligently query at $\mathbf{x}_t$ where the function is likely to be high. For this, we first use the posterior GP to construct an acquisition function φ_t which captures the value of the experiment at a point. Then we maximise φ_t to determine $\mathbf{x}_t$.

Gaussian process bandits and Bayesian optimisation (GPB/ BO) have been successfully applied in many applications such as tuning hyperparameters in learning algorithms (Snoek et al., 2012; Bergstra et al., 2011; Mahendran et al., 2012), robotics (Lizotte et al., 2007; Martinez-Cantin et al., 2007) and object tracking (Denil et al., 2012). However, all such successes have been in low (typically < 10) dimensions (Wang et al., 2013). Expensive high dimensional functions occur in several problems in fields such as computer vision (Yamins et al., 2013), antenna design (Hornby et al., 2006), computational astrophysics (Parkinson et al., 2006) and biology (Gonzalez et al., 2014). Scaling GPB/ BO methods to high dimensions for practical problems has been challenging. Even current theoretical results suggest that GPB/ BO is exponentially difficult in high dimensions without further assumptions (Srinivas et al., 2010; Bull, 2011). To our knowledge, the only approach to date has been to perform regular GPB/ BO on a low dimensional subspace. This works only under strong assumptions.

We identify two key challenges in scaling GPB/ BO to high dimensions. **The first is the statistical challenge in estimating the function**. Nonparametric regression is inherently difficult in high dimensions with known lower bounds depending exponentially in dimension (Györfi et al., 2002). The often exponential sample complexity for regression is invariably reflected in the regret bounds for GPB/ BO. **The second is the computational challenge in maximising** φ_t. Commonly used global optimisation heuristics used to maximise φ_t themselves require computation exponential in dimension. Any attempt to scale GPB/ BO to high dimensions must effectively address these two concerns.

Proceedings of the 32^{nd} *International Conference on Machine Learning*, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

In this work, we embark on this challenge by treating f as an *additive function* of mutually exclusive lower dimensional components. **Our contributions** in this work are:

1. We present the **Add-GP-UCB** algorithm for optimisation and bandits of an additive function. An attractive property is that we use an acquisition function which is easy to optimise in high dimensions.
2. In our theoretical analysis we bound the regret for **Add-GP-UCB**. We show that it has only linear dependence on the dimension D when f is additive[1].
3. Empirically we demonstrate that **Add-GP-UCB** outperforms naive BO on synthetic experiments, an astrophysical simulator and the Viola and Jones face detection problem. Furthermore **Add-GP-UCB** does well on several examples *when the function is not additive*.

A Matlab implementation of our methods is available online at `github.com/kirthevasank/add-gp-bandits`.

2. Related Work

GPB/ BO methods follow a family of GP based active learning methods which select the next experiment based on the posterior (Osborne et al., 2012; Ma et al., 2015; Kandasamy et al., 2015). In the GPB/ BO setting, common acquisition functions include Expected improvement (Mockus, 1994), probability of improvement (Jones et al., 1998), Thompson sampling (Thompson, 1933) and upper confidence bound (Auer, 2003). Of particular interest to us, is the Gaussian process upper confidence bound (**GP-UCB**). It was first proposed and analysed in the noisy setting by Srinivas et al. (2010) and extended to the noiseless case by de Freitas et al. (2012). Some literature studies variants, such as combining several acquisition functions (Hoffman et al., 2011) and querying in batches (Azimi et al., 2010).

To our knowledge, most literature for GPB/ BO in high dimensions are in the setting where the function varies only along a very low dimensional subspace (Chen et al., 2012; Wang et al., 2013; Djolonga et al., 2013). In these works, the authors do not encounter either challenge as they perform GPB/ BO in either a random or carefully selected lower dimensional subspace. However, assuming that the problem is an easy (low dimensional) one hiding in a high dimensional space is often too restrictive. Indeed, our experimental results confirm that such methods perform poorly on real applications when the assumptions are not met. While our additive assumption is strong in its own right, it is considerably more expressive. It is more general than the setting in Chen et al. (2012). Even though it does not contain the settings in Djolonga et al. (2013); Wang et al. (2013), unlike them, we still allow the function to vary along the entire domain.

Using an additive structure is standard in high dimensional regression literature both in the GP framework and otherwise. Hastie & Tibshirani (1990); Ravikumar et al. (2009) treat the function as a sum of one dimensional components. Our additive framework is more general. Duvenaud et al. (2011) assume a sum of functions of all combinations of lower dimensional coordinates. These literature argue that using an additive model has several advantages even if f is *not* additive. It is a well understood notion in statistics that when we only have a few samples, using a simpler model to fit our data may give us a better trade off for estimation error against approximation error. This observation is *crucial*: in many applications for Bayesian optimisation we are forced to work in the low sample regime since calls to the blackbox are expensive. Though the additive assumption is biased for nonadditive functions, it enables us to do well with only a few samples. While we have developed theoretical results only for additive f, empirically we show that our additive model outperforms naive GPB/ BO even when the underlying function is not additive.

Analyses of GPB/ BO methods focus on the query complexity of f which is the dominating cost in relevant applications. It is usually assumed that φ_t can be maximised to arbitrary precision at negligible cost. Common techniques to maximise φ_t include grid search, Monte Carlo and multistart methods (Brochu et al., 2010). In our work we use the Dividing Rectangles (DiRect) algorithm of Jones et al. (1993). While these methods are efficient in low dimensions they require exponential computation in high dimensions. It is widely acknowledged in the community that this is a critical bottleneck in scaling GPB/ BO to high dimensions (de Freitas, 2014). While we still work in the paradigm where evaluating f is expensive and characterise our theoretical results in terms of query complexity, we believe that assuming arbitrary computational power to optimise φ_t is too restrictive. For instance, in hyperparameter tuning the budget for determining the next experiment is dictated by the cost of the learning algorithm. In online advertising and robotic reinforcement learning we need to act in under a few seconds or real time.

In this manuscript, Section 3 formally details our problem and assumptions. We present **Add-GP-UCB** in Section 4 and our theoretical results in Section 4.3. All proofs are deferred to Appendix B. We summarize the regrets for **Add-GP-UCB** and **GP-UCB** in Table 1. In Section 5 we present the experiments.

[1] Post-publication it was pointed out to us that there was a bug in our analysis. We are working on resolving it and will post an update shortly. See Section 6 for more details.

Kernel	Squared Exponential	Matérn
GP-UCB on D^{th} order kernel	$\sqrt{D^{D+2}T(\log T)^{D+2}}$	$2^D\sqrt{D}T^{\frac{\nu+D(D+1)}{2\nu+D(D+1)}}\log T$
Add-GP-UCB on additive kernel	$\sqrt{d^dD^2T(\log T)^{d+2}}$	$2^dDT^{\frac{\nu+d(d+1)}{2\nu+d(d+1)}}\log T$

Table 1. Comparison of Cumulative Regret for **GP-UCB** and **Add-GP-UCB** for the Squared Exponential and Matérn kernels.

3. Problem Statement & Set up

We wish to maximise a function $f : \mathcal{X} \rightarrow \mathbb{R}$ where $\mathcal{X}$ is a *rectangular* region in $\mathbb{R}^D$. We will assume w.l.o.g $\mathcal{X} = [0,1]^D$. f may be nonconvex and gradient information is not available. We can interact with f only by querying at some $x \in \mathcal{X}$ and obtain a noisy observation $y = f(x)+\epsilon$. Let an optimum point be $\mathbf{x}_* = \text{argmax}_{x\in\mathcal{X}} f(x)$. Suppose at time t we choose to query at $\mathbf{x}_t$. Then we incur *instantaneous regret* $r_t = f(\mathbf{x}_*) - f(\mathbf{x}_t)$. In the bandit setting, we are interested in the *cumulative regret* $R_T = \sum_{t=1}^T r_t = \sum_{t=1}^T f(\mathbf{x}_*) - f(\mathbf{x}_t)$, and in the optimisation setting we are interested in the *simple regret* $S_T = \min_{t\leq T} r_t = f(\mathbf{x}_*) - \max_{\mathbf{x}_t} f(\mathbf{x}_t)$. For a bandit algorithm, a desirable property is to have *no regret*: $\lim_{T\to\infty} \frac{1}{T}R_T = 0$. Since $S_T \leq \frac{1}{T}R_T$, any such procedure is also a consistent procedure for optimisation.

Key structural assumption: In order to make progress in high dimensions, we will assume that f decomposes into the following additive form,

$$f(x) = f^{(1)}(x^{(1)}) + f^{(2)}(x^{(2)}) + \cdots + f^{(M)}(x^{(M)}). \quad (1)$$

Here each $x^{(j)} \in \mathcal{X}^{(j)} = [0,1]^{d_j}$ are lower dimensional components. We will refer to the $\mathcal{X}^{(j)}$'s as "groups" and the grouping of different dimensions into these groups $\{\mathcal{X}^{(j)}\}_{j=1}^M$ as the "decomposition". The groups are *disjoint* – i.e. if we treat the elements of the vector x as a set, $x^{(i)} \cap x^{(j)} = \varnothing$. We are primarily interestd in the case when D is very large and the group dimensionality is bounded: $d_j \leq d \ll D$. We have $D \asymp dM \geq \sum_j d_j$. Paranthesised superscripts index the groups and a union over the groups denotes the reconstruction of the whole from the groups (e.g. $x = \bigcup_j x^{(j)}$ and $\mathcal{X} = \bigcup_j \mathcal{X}^{(j)}$). $\mathbf{x}_t$ denotes the point chosen by the algorithm for querying at time t. We will ignore $\log D$ terms in $\mathcal{O}(\cdot)$ notation. Our theoretical analysis assumes that the decomposition is known but we also present a modified algorithm to handle unknown decompositions and non-additive functions.

Some smoothness assumptions on f are warranted to make the problem tractable. A standard in the Bayesian paradigm is to assume f is sampled from a Gaussian Process (Rasmussen & Williams, 2006) with a covarince kernel $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ and that $\epsilon \sim \mathcal{N}(0, \eta^2)$. Two commonly used kernels are the squared exponential (SE) $\kappa_{\sigma,h}$ and the Matérn $\kappa_{\nu,h}$ kernels with parameters (σ, h) and (ν, h) respectively. Writing $r = \|x - x'\|_2$, they are defined as

$$\kappa_{\sigma,h}(x,x') = \sigma \exp\left(\frac{-r^2}{2h^2}\right), \quad (2)$$

$$\kappa_{\nu,h}(x,x') = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}r}{h}\right)^{\nu} B_\nu\left(\frac{\sqrt{2\nu}r}{h}\right). \quad (3)$$

Here Γ, B_ν are the Gamma and modified Bessel functions. A principal convenience in modelling our problem via a GP is that posterior distributions are analytically tractable.

In keeping with this, we will assume that each $f^{(j)}$ is sampled from a GP, $\mathcal{GP}(\mu^{(j)}, \kappa^{(j)})$ where the $f^{(j)}$'s are independent. Here, $\mu^{(j)} : \mathcal{X}^{(j)} \rightarrow \mathbb{R}$ is the mean and $\kappa^{(j)} : \mathcal{X}^{(j)} \times \mathcal{X}^{(j)} \rightarrow \mathbb{R}$ is the covariance for $f^{(j)}$. W.l.o.g let $\mu^{(j)} = \mathbf{0}$ for all j. This implies that f itself is sampled from a GP with an additive kernel $\kappa(x,x') = \sum_j \kappa^{(j)}(x^{(j)}, x^{(j)'})$. We state this formally for nonzero mean as we will need it for the ensuing discussion.

Observation 1. *Let f be defined as in Equation (1), where $f^{(j)} \sim \mathcal{GP}(\mu^{(j)}(x), \kappa^{(j)}(x^{(i)}, x^{(j)'}))$. Let $y = f(x) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \eta^2)$. Denote $\delta(x,x') = 1$ if $x = x'$, and 0 otherwise. Then $y \sim \mathcal{GP}(\mu(x), \kappa(x,x') + \eta^2\delta(x,x'))$ where*

$$\mu(x) = \mu^{(1)}(x^{(1)}) + \cdots + \mu^{(M)}(x^{(M)}) \quad (4)$$
$$\kappa(x,x') = \kappa^{(1)}(x^{(1)}, x^{(1)'}) + \cdots + \kappa^{(M)}(x^{(M)}, x^{(M)'}).$$

We will call a kernel such as $\kappa^{(j)}$ which acts only on d variables a d^{th} order kernel. A kernel which acts on all the variables is a D^{th} order kernel. Our kernel for f is a sum of M at most d^{th} order kernels which, we will show, is statistically simpler than a D^{th} order kernel.

We conclude this section by looking at some seemingly straightforward approaches to tackle the problem. The first natural question is of course why not directly run **GP-UCB** using the additive kernel? Since it is simpler than a D^{th} order kernel we can expect statistical gains. While this is true, it still requires optimising φ_t in D dimensions to determine the next point which is expensive.

Alternatively, for an additive function, we could adopt a sequential approach where we use $1/M$ fraction of our query budget to maximise the first group by keeping the rest of the coordinates constant. Then we proceed to the second

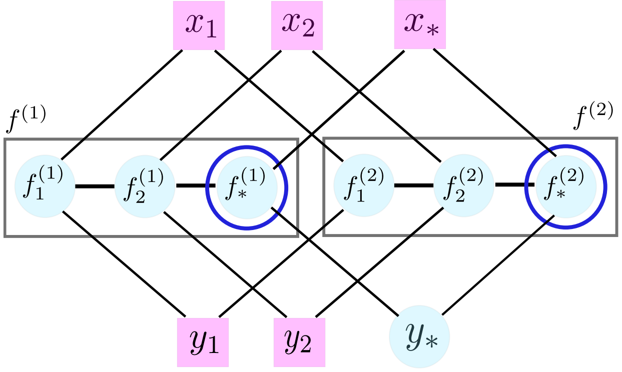


Figure 1. Illustration of the additive GP model for 2 observations where $M = 2$ in (1). The squared variables are observed while the circled variables are not. For brevity we have denoted $f_i^{(j)} = f^{(j)}(x_i^{(j)})$ for $i = 1, 2, *$. We wish to infer the posterior distributions of the individual GPs $f^{(j)}(x_*^{(j)})$ (outlined in blue).

group and so on. While optimising a d dimensional acquisition function is easy, this approach is not desirable for several reasons. First, it will not be an anytime algorithm as we will have to pre-allocate our query budget to maximise each group. Once we proceed to a new group we cannot come back and optimise an older one. Second, such an approach places too much faith in the additive assumption. We will only have explored M d-dimensional hyperplanes in the entire space. Third, it is not suitable as a bandit algorithm as we suffer high regret until we get to the last group. We further elaborate on the deficiencies of this and other sequential approaches in Appendix A.2.

4. Algorithm

Under an additive assumption, our algorithm has two components. First, we obtain the posterior GP for each $f^{(j)}$ using the query-value pairs until time t. Then we maximise a d dimensional **GP-UCB**-like acquisition function on *each* GP to construct the next query point. Since optimising φ_t depends exponentially in dimension, this is cheaper than optimising one acquisition on the combined GP.

4.1. Inference on Additive GPs

Typically in GPs, given noisy labels, $Y = \{y_1, \dots, y_n\}$ at points $X = \{x_1, \dots, x_n\}$, we are interested in inferring the posterior distribution for $f_* = f(x_*)$ at a new point x_*. In our case though, we will be primarily interested in the distribution of $f_*^{(j)} = f^{(j)}(x_*^{(j)})$ conditioned on X, Y. We have illustrated this graphically in Figure 1. The joint distribution of $f_*^{(j)}$ and Y can be written as

$$\begin{pmatrix} f_*^{(j)} \\ Y \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \kappa^{(j)}(x_*^{(j)}, x_*^{(j)}) & \kappa^{(j)}(x_*^{(j)}, X^{(j)}) \\ \kappa^{(j)}(X^{(j)}, x_*^{(j)}) & \kappa(X, X) + \eta^2 I_n \end{bmatrix}\right).$$

The p^{th} element of $\kappa^{(j)}(X^{(j)}, x_*^{(j)}) \in \mathbb{R}^n$ is $\kappa(x_p^{(j)}, x_*^{(j)})$ and the $(p,q)^{\text{th}}$ element of $\kappa(X, X) \in \mathbb{R}^{n\times n}$ is $\kappa(x_p, x_q)$. We have used the fact $\mathrm{Cov}(f_*^{(i)}, y_p) = \mathrm{Cov}(f_*^{(i)}, \sum_j f^{(j)}(x_p^{(j)}) + \eta^2\epsilon) = \mathrm{Cov}(f_*^{(i)}, f^{(i)}(x_p^{(i)})) = \kappa^{(i)}(x_*^{(i)}, x_p^{(i)})$ as $f^{(j)} \perp f^{(i)}, \forall i \neq j$. By writing $\Delta = \kappa(X, X) + \eta^2 I_n \in \mathbb{R}^{n\times n}$, the posterior for $f_*^{(j)}$ is,

$$\begin{aligned} f_*^{(j)} | x_*, X, Y \sim \mathcal{N}\big(&\kappa^{(j)}(x_*^{(j)}, X^{(j)})\Delta^{-1}Y, \\ &\kappa^{(j)}(x_*^{(j)}, x_*^{(j)}) - \kappa^{(j)}(x_*^{(j)}, X^{(j)})\Delta^{-1}\kappa^{(j)}(X, x^{(j)})\big) \end{aligned} \qquad (5)$$

4.2. The Add-GP-UCB Algorithm

In GPB/ BO algorithms, at each time step t we maximise an acquisition function φ_t to determine the next point: $\mathbf{x}_t = \mathrm{argmax}_{x\in\mathcal{X}}\, \varphi_t(x)$. The acquisition function is itself constructed using the posterior GP. The **GP-UCB** acquisition function, which we focus on here is,

$$\varphi_t(x) = \mu_{t-1}(x) + \beta_t^{1/2}\sigma_{t-1}(x).$$

Intuitively, the μ_{t-1} term in the **GP-UCB** objective prefers points where f is known to be high, the σ_{t-1} term prefers points where we are uncertain about f and $\beta_t^{1/2}$ negotiates the tradeoff. The former contributes to the "exploitation" facet of our problem, in that we wish to have low instantaneous regret. The latter contributes to the "exploration" facet since we also wish to query at regions we do not know much about f lest we miss out on regions where f is high. We provide a brief summary of **GP-UCB** and its theoretical properties in Appendix A.1.

As we have noted before, maximising φ_t which is typically multimodal to obtain $\mathbf{x}_t$ is itself a difficult problem. In any grid search or branch and bound methods such as DiRect, maximising a function to within ζ accuracy, requires $\mathcal{O}(\zeta^{-D})$ calls to φ_t. Therefore, for large D maximising φ_t is extremely difficult. In practical settings, especially in situations where we are computationally constrained, this poses serious limitations for GPB/ BO as we may not be able to optimise φ_t to within a desired accuracy.

Fortunately, in our setting we can be more efficient. We propose an alternative acquisition function which applies to an additive kernel. We define the *Additive Gaussian Process Upper Confidence Bound* (**Add-GP-UCB**) to be

$$\widetilde{\varphi}_t(x) = \mu_{t-1}(x) + \beta_t^{1/2}\sum_{j=1}^{M}\sigma_{t-1}^{(j)}(x^{(j)}). \qquad (6)$$

We immediately see that we can write $\widetilde{\varphi}_t$ as a sum of functions on orthogonal domains: $\widetilde{\varphi}_t(x) = \sum_j \widetilde{\varphi}_t^{(j)}(x^{(j)})$ where $\widetilde{\varphi}_t^{(j)}(x^{(j)}) = \mu_{t-1}^{(j)}(x^{(j)}) + \beta_t^{1/2}\sigma_{t-1}^{(j)}(x^{(j)})$. This means that $\widetilde{\varphi}_t$ can be maximised by maximising each $\widetilde{\varphi}_t^{(j)}$ separately on $\mathcal{X}^{(j)}$. As we need to solve M at most d dimensional optimisation problems, it requires only $\mathcal{O}(M^{d+1}\zeta^{-d})$ calls to the utility function in total – far more favourable than maximising φ_t.

Since the cost for maximising the acquisition function is a key theme in this paper let us delve into this a bit more. One call to φ_t requires $\mathcal{O}(Dt^2)$ effort. For $\widetilde{\varphi}_t$ we need M calls each requiring $\mathcal{O}(d_j t^2)$ effort. So both φ_t and $\widetilde{\varphi}_t$ require the same effort in this front. For φ_t, we need to know the posterior for only f whereas for $\widetilde{\varphi}_t$ we need to know the posterior for each $f^{(j)}$. However, the brunt of the work in obtaining the posterior is the $\mathcal{O}(t^3)$ effort in inverting the $t\times t$ matrix Δ in (5) which needs to be done for both φ_t and $\widetilde{\varphi}_t$. For $\widetilde{\varphi}_t$, we can obtain the inverse once and reuse it M times, so the cost of obtaining the posterior is $\mathcal{O}(t^3+Mt^2)$. Since the number of queries needed will be super linear in D and hence M, the t^3 term dominates. Therefore obtaining each posterior $f^{(j)}$ is only marginally more work than obtaining the posterior for f. Any difference here is easily offset by the cost for maximising the acquisition function.

The question remains then if maximising $\widetilde{\varphi}_t$ would result in low regret. Since φ_t and $\widetilde{\varphi}_t$ are neither equivalent nor have the same maximiser it is not immediately apparent that this should work. Nonetheless, intuitively this seems like a reasonable scheme since the $\sum_j \sigma_{t-1}^{(j)}$ term captures some notion of the uncertainty and contributes to exploration. In Theorem 5 we show that this intuition is reasonable – maximising $\widetilde{\varphi}_t$ achieves the *same* rates as φ_t for cumulative and simple regrets if the kernel is additive.

We summarise the resulting algorithm in Algorithm 1. In brief, at time step t, we obtain the posterior distribution for $f^{(j)}$ and maximise $\widetilde{\varphi}_t^{(j)}$ to determine the coordinates $\mathbf{x}_t^{(j)}$. We do this for each j and then combine them to obtain $\mathbf{x}_t$.

Algorithm 1 Add-GP-UCB

```
Input: Kernels $\kappa^{(1)},\dots,\kappa^{(M)}$, Decomposition $(\mathcal{X}^{(j)})_{j=1}^M$
  • $\mathcal{D}_0 \leftarrow \varnothing$,
  • for $j = 1,\dots,M$, $(\mu_0^{(j)}, \kappa_0^{(j)}) \leftarrow (\mathbf{0}, \kappa^{(j)})$.
  • for $t = 1, 2, \dots$
      1. for $j = 1,\dots,M$,
         $\mathbf{x}_t^{(j)} \leftarrow \mathrm{argmax}_{z\in\mathcal{X}^{(j)}}\ \mu_{t-1}^{(j)}(z) + \sqrt{\beta_t}\sigma_{t-1}^{(j)}(z)$
      2. $\mathbf{x}_t \leftarrow \bigcup_{j=1}^M \mathbf{x}_t^{(j)}$.
      3. $\mathbf{y}_t \leftarrow$ Query $f$ at $\mathbf{x}_t$.
      4. $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, \mathbf{y}_t)\}$.
      5. Perform Bayesian posterior updates conditioned
         on $\mathcal{D}_t$ to obtain $\mu_t^{(j)}, \sigma_t^{(j)}$ for $j = 1,\dots,M$.
```

4.3. Main Theoretical Results

Now, we present our main theoretical contributions. We bound the regret for **Add-GP-UCB** under different kernels. Following Srinivas et al. (2010), we first bound the statistical difficulty of the problem as determined by the kernel. We show that under additive kernels the problem is much easier than when using a full D^{th} order kernel. Next, we show that the **Add-GP-UCB** algorithm is able to exploit the additive structure and obtain the same rates as **GP-UCB**. The advantage to using **Add-GP-UCB** is that it is much easier to optimise the acquisition function. For our analysis, we will need Assumption 2 and Definition 3.

Assumption 2. *Let f be sampled from a GP with kernel κ. $\kappa(\cdot, x)$ is L-Lipschitz for all x. Further, the partial derivatives of f satisfies the following high probability bound. There exists constants $a, b > 0$ such that,*

$$\mathbb{P}\left(\sup_x \left|\frac{\partial f(x)}{\partial x_i}\right| > J\right) \le a e^{-(J/b)^2}.$$

The Lipschitzian condition is fairly mild and the latter condition holds for four times differentiable stationary kernels such as the SE and Matérn kernels for $\nu > 2$ (Ghosal & Roy, 2006). Srinivas et al. (2010) showed that the statistical difficulty of GPB/ BO is determined by the *Maximum Information Gain* as defined below. We bound this quantity for additive SE and Matérn kernels in Theorem 4. This is our first main theorem.

Definition 3. *(Maximum Information Gain) Let $f \sim \mathcal{GP}(\mu, \kappa)$, $y_i = f(x_i) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \eta^2)$. Let $A = \{x_1, \dots, x_T\} \subset \mathcal{X}$ be a finite subset, f_A denote the function values at these points and y_A denote the noisy observations. Let I be the Shannon Mutual Information. The Maximum Information Gain between y_A and f_A is*

$$\gamma_T = \max_{A\subset\mathcal{X}, |A|=T} I(y_A; f_A).$$

Theorem 4. *Assume that the kernel κ has the additive form of (4), and that each $\kappa^{(j)}$ satisfies Assumption 2. W.l.o.g assume $\kappa(x, x') = 1$. Then,*

1. *If each $\kappa^{(j)}$ is a d_j^{th} order squared exponential kernel (2) where $d_j \le d$, then $\gamma_T \in \mathcal{O}(Dd^d(\log T)^{d+1})$.*
2. *If each $\kappa^{(j)}$ is a d_j^{th} order Matérn kernel (3) where $d_j \le d$ and $\nu > 2$, then $\gamma_T \in \mathcal{O}(D2^dT^{\frac{d(d+1)}{2\nu+d(d+1)}}\log(T))$.*

We use bounds on the eigenvalues of the SE and Matérn kernels from Seeger et al. (2008) and a result from Srinivas et al. (2010) which bounds the information gain via the eigendecay of the kernel. We bound the eigendecay of the sum κ via M and the eigendecay of a single $\kappa^{(j)}$. The complete proof is given in Appendix B.1. The important observation is that the dependence on D is linear for an additive kernel. In contrast, for a D^{th} order kernel this is exponential (Srinivas et al., 2010).

Next, we present our second main theorem which bounds the regret for **Add-GP-UCB** for an additive kernel as given in Equation 4.

Theorem 5. *Suppose f is constructed by sampling $f^{(j)} \sim \mathcal{GP}(\mathbf{0}, \kappa^{(j)})$ for $j = 1, \ldots, M$ and then adding them. Let all kernels $\kappa^{(j)}$ satisfy assumption 2 for some L, a, b. Further, we maximise the acquisition function $\widetilde{\varphi}_t$ to within $\zeta_0 t^{-1/2}$ accuracy at time step t. Pick $\delta \in (0,1)$ and choose*

$$\beta_t = 2\log\left(\frac{M\pi^2 t^2}{2\delta}\right) + 2d\log\left(Dt^3\right) \in \mathcal{O}\left(d\log t\right).$$

*Then, **Add-GP-UCB** attains cumulative regret $R_T \in \mathcal{O}\left(\sqrt{D\gamma_T T\log T}\right)$ and hence simple regret $S_T \in \mathcal{O}\left(\sqrt{D\gamma_T \log T/T}\right)$. Precisely, with probability $> 1-\delta$,*

$$\forall T \geq 1, \quad R_T \leq \sqrt{8C_1\beta_T M T\gamma_t} + 2\zeta_0\sqrt{T} + C_2.$$

where $C_1 = 1/\log(1+\eta^{-2})$ and C_2 is a constant depending on a, b, D, δ, L and η.

Part of our proof uses ideas from Srinivas et al. (2010). We show that $\sum_j \beta_t \sigma_{t-1}^{(j)}(\cdot)$ forms a credible interval for $f(\cdot)$ about the posterior mean $\mu_t(\cdot)$ for an additive kernel in **Add-GP-UCB**. We relate the regret to this confidence set using a covering argument. We also show that our regret doesn't suffer severely if we only approximately optimise the acquisition provided that the accuracy improves at rate $\mathcal{O}(t^{-1/2})$. For this we establish smoothness of the posterior mean. The correctness of the algorithm follows from the fact that **Add-GP-UCB** can be maximised by individually maximising $\widetilde{\varphi}_t^{(j)}$ on each $\mathcal{X}^{(j)}$. The complete proof is given in Appendix B.2. When we combine the results in Theorems 4 and 5 we obtain the rates given in Table 1[2].

One could consider alternative lower order kernels – one candidate is the sum of all possible d^{th} order kernels (Duvenaud et al., 2011). Such a kernel would arguably allow us to represent a larger class of functions than our kernel in (4). If, for instance, we choose each of them to be a SE kernel, then it can be shown that $\gamma_T \in \mathcal{O}(D^d d^{d+1}(\log T)^{d+1})$. Even though this is worse than our kernel in poly(D) factors, it is still substantially better than using a D^{th} order kernel. However, maximising the corresponding utility function, either of the form φ_t or $\widetilde{\varphi}_t$, is still a D dimensional problem. We reiterate that what renders our algorithm attractive in large D is not just the statistical gains due to the simpler kernel. It is also the fact that our acquisition function can be efficiently maximised.

4.4. Practical Considerations

Our practical implementation differs from our theoretical analysis in the following aspects.

Choice of β_t: β_t as specified by Theorems 5, usually tends to be conservative in practice (Srinivas et al., 2010). For good empirical performance a more aggressive strategy is required. In our experiments, we set $\beta_t = 0.2d\log(2t)$ which offered a good tradeoff between exploration and exploitation. Note that this captures the correct dependence on D, d and t in Theorems 5 and 6.

Data dependent prior: Our analysis assumes that we know the GP kernel of the prior. In reality this is rarely the case. In our experiments, we choose the hyperparameters of the kernel by maximising the GP marginal likelihood (Rasmussen & Williams, 2006) every N_{cyc} iterations.

Initialisation: Marginal likelihood based kernel tuning can be unreliable with few data points. This is a problem in the first few iterations. Following the recommendations in Bull (2011) we initialise **Add-GP-UCB** (and **GP-UCB**) using N_{init} points selected uniformly at random.

Decomposition & Non-additive functions: If f is additive and the decomposition is known, we use it directly. But it may not always be known or f may not be additive. Then, we could treat the decomposition as a hyperparameter of the additive kernel and maximise the marginal likelihood w.r.t the decomposition. However, given that there are $D!/d!^M M!$ possible decompositions, computing the marginal likelihood for all of them is infeasible. We circumvent this issue by randomly selecting a few $(\mathcal{O}(D))$ decompositions and choosing the one with the largest marginal likelihood. Intuitively, if the function is not additive, with such a "partial maximisation" we can hope to capture some existing marginal structure in f. At the same time, even an exhaustive maximisation will not do much better than a partial maximisation if there is no additive structure. Empirically, we found that partially optimising for the decomposition performed slightly better than using a fixed decomposition or a random decomposition at each step. We incorporate this procedure for finding an appropriate decomposition as part of the kernel hyper parameter learning procedure every N_{cyc} iterations.

How do we choose (d, M) when f is not additive? If d is large we allow for richer class of functions, but risk high variance. For small d, the kernel is too simple and we have high bias but low variance – further optimising $\widetilde{\varphi}_t$ is easier. In practice we found that our procedure was fairly robust for reasonable choices of d. Yet this is an interesting theoretical question. We also believe it is a difficult one. Using the marginal likelihood alone will not work as the optimal choice of d also depends on the computational budget for optimising $\widetilde{\varphi}_t$. We hope to study this question in future work. For now, we give some recommendations at the end. Our modified algorithm with these practical considerations is given below. Observe that in this specification if we use $d = D$ we have the original **GP-UCB** algorithm.

[2] See Footnote 1.

Algorithm 2 **Practical-Add-GP-UCB**

```
Input: $N_{init}$, $N_{cyc}$, $d$, $M$
  • $\mathcal{D}_0 \leftarrow N_{init}$ points chosen uniformly at random.
  • for $t = 1, 2, \ldots$
      1. if ($t \mod N_{cyc} = 0$), Learn the kernel hyper parameters and the decomposition $\{\mathcal{X}_j\}$ by maximising the GP marginal likelihood.
      2. Perform steps 1-3 in Algorithm 1 with $\beta_t = 0.2d \log 2t$.
      3. $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, \mathbf{y}_t)\}$.
      4. Perform Bayesian posterior updates conditioned on $\mathcal{D}_t$ to obtain $\mu_t^{(j)}, \sigma_t^{(j)}$ for $j = 1, \ldots, M$.
```

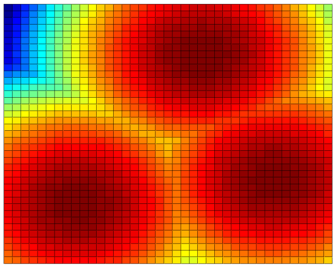


Figure 2. Illustration of the trimodal function $f_{d'}$ in $d' = 2$.

5. Experiments

To demonstrate the efficacy of **Add-GP-UCB** over **GP-UCB** we optimise the acquisition function under a constrained budget. Following, Brochu et al. (2010) we use DiRect to maximise $\varphi_t, \widetilde{\varphi}_t$. We compare **Add-GP-UCB** against **GP-UCB**, random querying (RAND) and DiRect[3]. On the real datasets we also compare it to the Expected Improvement (**GP-EI**) acquisition function which is popular in BO applications and the method of Wang et al. (2013) which uses a random projection before applying BO (REMBO). We have multiple instantiations of **Add-GP-UCB** for different values for (d, M). For optimisation, we perform comparisons based on the simple regret S_T and for bandits we use the time averaged cumulative regret R_T/T.

For all GPB/ BO methods we set $N_{init} = 10$, $N_{cyc} = 25$ in all experiments. Further, for the first 25 iterations we set the bandwidth to a small value (10^{-5}) to encourage an explorative strategy. We use SE kernels for each additive kernels and use the same scale σ and bandwidth h hyperparameters for all the kernels. Every 25 iterations we maximise the marginal likelihood with respect to these 2 hyperparameters in addition to the decomposition.

In contrast to existing literature in the BO community, we found that the UCB acquisitions outperformed **GP-EI**. One possible reason may be that under a constrained budget, UCB is robust to imperfect maximisation (Theorem 5) whereas **GP-EI** may not be. Another reason may be our choice of constants in UCB (Section 4.4).

5.1. Simulations on Synthetic Data

First we demonstrate our technique on a series of synthetic examples. For this we construct additive functions for different values for the maximum group size d' and the number of groups M'. We use the prime to distinguish it from **Add-GP-UCB** instantiations with different combinations of (d, M) values. The d' dimensional function $f_{d'}$ is,

$$f_{d'}(x) = \log\left(0.1\frac{1}{h_{d'}^{d'}} \exp\left(\frac{\|x - v_1\|^2}{2h_{d'}^2}\right) + \right. \quad (7)$$
$$\left. 0.1\frac{1}{h_{d'}^{d'}} \exp\left(\frac{\|x - v_2\|^2}{2h_{d'}^2}\right) + 0.8\frac{1}{h_{d'}^{d'}} \exp\left(\frac{\|x - v_3\|^2}{2h_{d'}^2}\right)\right)$$

where v_1, v_2, v_3 are fixed d' dimensional vectors and $h_{d'} = 0.01{d'}^{0.1}$. Then we create M' groups of coordinates by randomly adding d' coordinates into each group. On each such group we use $f_{d'}$ and then add them up to obtain the composite function f. Precisely,

$$f(x) = f_{d'}(x^{(1)}) + \cdots + f_{d'}(x^{(M)})$$

The remaining $D - d'M'$ coordinates do not contribute to the function. Since $f_{d'}$ has 3 modes, f will have $3^{M'}$ modes. We have illustrated $f_{d'}$ for $d' = 2$ in Figure 2.

In the synthetic experiments we use an instantiation of **Add-GP-UCB** that knows the decomposition–i.e. $(d, M) = (d', M')$ and the grouping of coordinates. We refer to this as **Add-**$\star$. For the rest we use a (d, M) decomposition by creating M groups of size at most d and find a good grouping by partially maximising the marginal likelihood (Section 4.4). We refer to them as **Add-**d/M.

For **GP-UCB** we allocate a budget of $\min(5000, 100D)$ DiRect function evaluations to optimise the acquisition function. For all **Add-**d/M methods we set it to 90% of this amount[4] to account for the additional overhead in posterior inference for each $f^{(j)}$. Therefore, in our $10D$ problem we maximise φ_t with $\beta_t = 2\log(2t)$ with 1000 DiRect evaluations whereas for **Add-**$2/5$ we maximise each $\widetilde{\varphi}_t^{(j)}$ with $\beta_t = 0.4\log(2t)$ with 180 evaluations.

The results are given in Figures 3 and 4. We refer to each example by the configuration of the additive function–its (D, d', M') values. In the $(10, 3, 3)$ example **Add-**$\star$ does

[3] There are several optimisation methods based on simulated annealing, cross entropy and genetic algorithms. We use DiRect since its easy to configure and known to work well in practice.

[4] While the 90% seems arbitrary, in our experiments this was hardly a factor as the cost was dominated by the inversion of Δ.

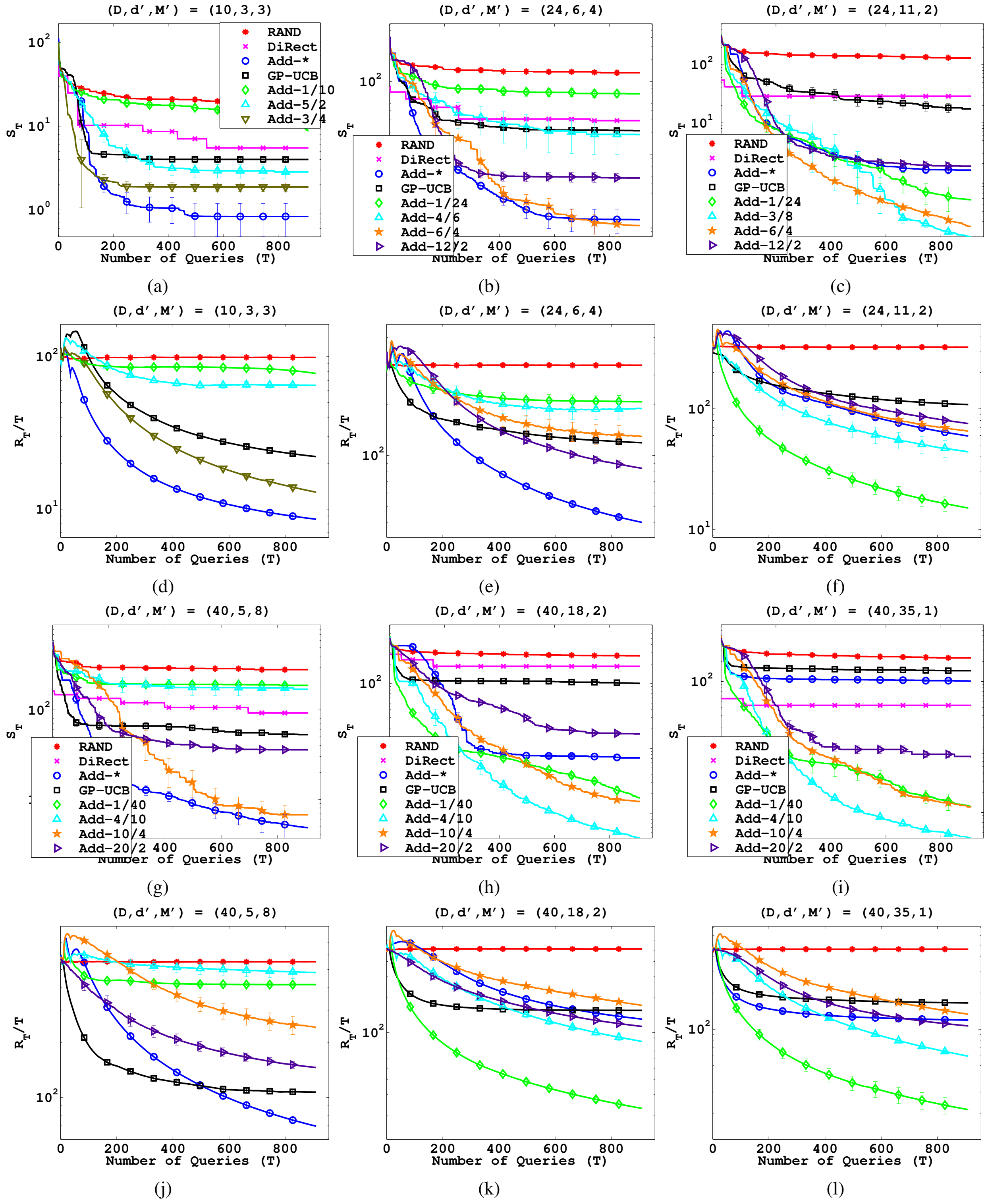
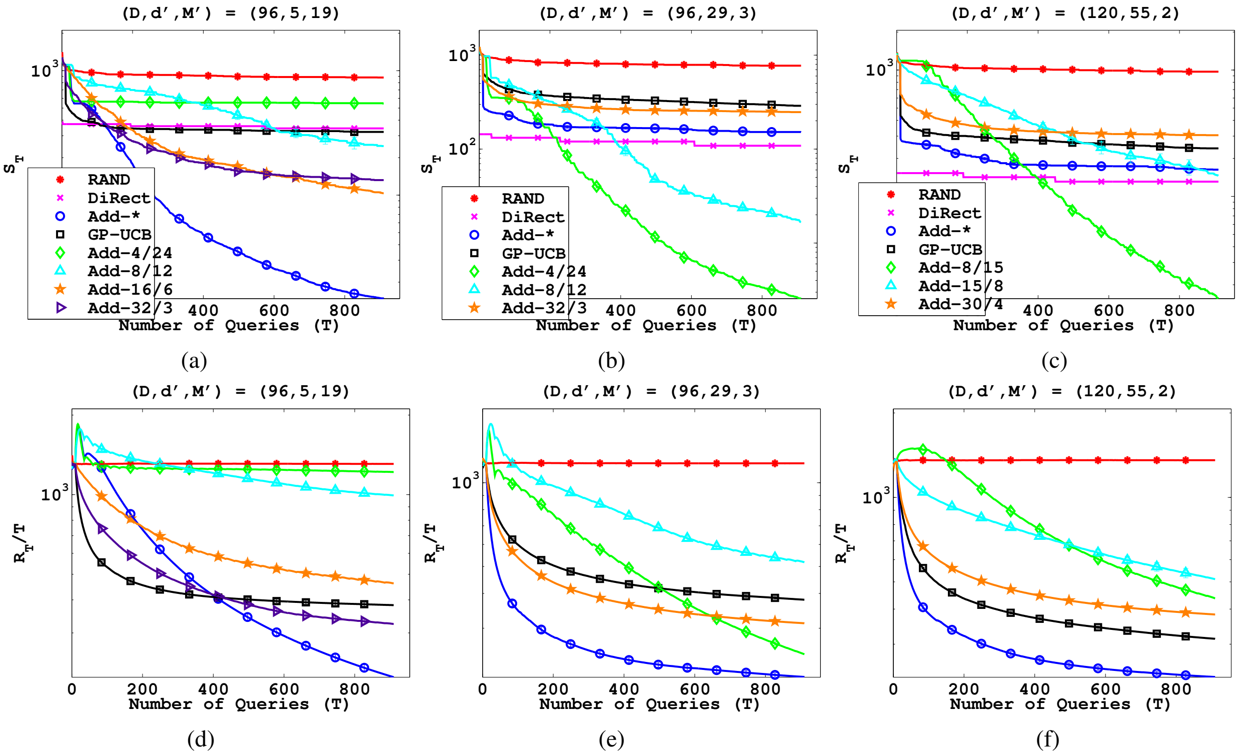


Figure 3. Results on the synthetic datasets. In all images the x-axis is the number of queries and the y-axis is the regret in log scale. We have indexed each experiment by their (D, d', M') values. The first row is S_T for the experiments with (D, d', M') set to $(10, 3, 3), (24, 6, 4), (24, 11, 2)$ and the second row is R_T/T for the same experiments. The third row is S_T for $(40, 5, 8), (40, 18, 2), (40, 35, 1)$ and the fourth row is the corresponding R_T/T. In some figures, the error bars are not visible since they are small and hidden by the bullets. All figures were produced by averaging over 20 runs.



(a) (b) (c)

(d) (e) (f)

Figure 4. More results on synthetic experiments. The simple regret S_T (first row) and cumulative regret R_T/T (second row) for functions with (D, d', M') set to $(96, 5, 19)$, $(96, 29, 3)$, $(120, 55, 2)$ respectively. Read the caption under Figure 3 for more details.

best since it knows the correct model and the acquisition function can be maximised within the budget. However **Add**-$3/4$ and **Add**-$5/2$ models do well too and outperform **GP-UCB**. **Add**-$1/10$ performs poorly since it is statistically not expressive enough to capture the true function. In the $(24, 11, 2)$, $(40, 18, 2)$, $(40, 35, 1)$, $(96, 29, 3)$ and $(120, 55, 2)$ examples **Add**-$\star$ outperforms **GP-UCB**. However, it is not competitive with the **Add**-d/M for small d. Even though **Add**-$\star$ knew the correct decomposition, there are two possible failure modes since d' is large. The kernel is complex and the estimation error is very high in the absence of sufficient data points. In addition, optimising the acquisition is also difficult. This illustrates our previous argument that using an additive kernel can be advantageous even if the function is not additive or the decomposition is not known. In the $(24, 6, 4)$, $(40, 5, 8)$ and $(96, 5, 19)$ examples **Add**-$\star$ performs best as d' is small enough. But again, almost all **Add**-d/M instantiations outperform **GP-UCB**. In contrast to the small D examples, for large D, **GP-UCB** and **Add**-d/M with large d perform worse than DiRect. This is probably because our budget for maximising φ_t is inadequate to optimise the acquisition function to sufficient accuracy. For some of the large D examples the cumulative regret is low for **Add-GP-UCB** and **Add**-d/M with large d. This is probably since they have already started exploiting where as the **Add**-d/M with small d methods are still exploring. We posit that if we run for more iterations we will be able to see the improvements.

5.2. SDSS Astrophysical Dataset

Here we used Galaxy data from the Sloan Digital Sky Survey (SDSS). The task is to find the maximum likelihood estimators for a simulation based astrophysical likelihood model. Data and software for computing the likelihood are taken from Tegmark et al (2006). The software itself takes in only 9 parameters but we augment this to 20 dimensions to emulate the fact that in practical astrophysical problems we may not know the true parameters on which the problem is dependent. This also allows us to effectively demonstrate the superiority of our methods over alternatives. Each query to this likelihood function takes about 2-5 seconds. In order to be wall clock time competitive with RAND and DiRectwe use only 500 evaluations for **GP-UCB**, **GP-EI** and REMBO and 450 for **Add**-d/M to maximise the acquisition function.

We have shown the Maximum value obtained over 400 iterations of each algorithm in Figure 5(a). Note that RAND outperforms DiRect here since a random query strategy is effectively searching in 9 dimensions. Despite this advantage to RAND all BO methods do better. Moreover, despite the fact that the function may not be additive, all

Add-d/M methods outperform **GP-UCB**. Since the function only depends on 9 parameters we use REMBO with a 9 dimensional projection. Yet, it is not competitive with the **Add**-d/M methods. Possible reasons for this may include the scaling of the parameter space by $\sqrt{d}$ in REMBO and the imperfect optimisation of the acquisition function. Here **Add**-$5/4$ performs slightly better than the rest since it seems to have the best tradeoff between being statistically expressive enough to capture the function while at the same time be easy enough to optimise the acquisition function within the allocated budget.

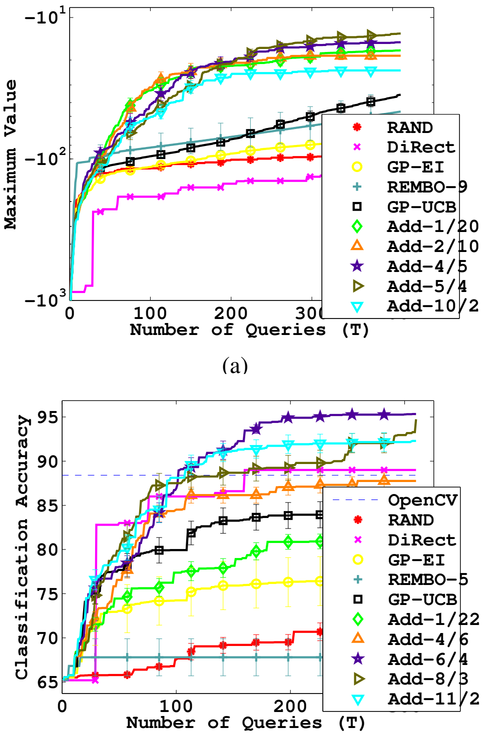
5.3. Viola & Jones Face Detection

The Viola & Jones (VJ) Cascade Classifier (Viola & Jones, 2001) is a popular method for face detection in computer vision based on the Adaboost algorithm. The K-cascade has K weak classifiers which outputs a score for any given image. When we wish to classify an image we pass that image through each classifier. If at any point the score falls below a certain threshold the image is classified as negative. If the image passes through all classifiers then it is classified as positive. The threshold values at each stage are usually pre-set based on prior knowledge. There is no reason to believe that these threshold values are optimal. In this experiment we wish to find an optimal set of values for these thresholds by optimising the classification accuracy over a training set.

For this task, we use 1000 images from the Viola & Jones face dataset containing both face and non-face images. We use the implementation of the VJ classifier that comes with OpenCV (Bradski & Kaehler, 2008) which uses a 22-stage cascade and modify it to take in the threshold values as a parameter. As our domain $\mathcal{X}$ we choose a neighbourhood around the configuration given in OpenCV. Each function call takes about 30-40 seconds and is the the dominant cost in this experiment. We use 1000 DiRect evaluations to optimise the acquisition function for **GP-UCB**, **GP-EI** and REMBO and 900 for the **Add**-d/M instantiations. Since we do not know the structure of the function we use REMBO with a 5 dimensional projection. The results are given in Figure 5(b). Not surprisingly, REMBO performs worst as it is only searching on a 5 dimensional space. Barring **Add**-$1/22$ all other instantiations perform better than **GP-UCB** and **GP-EI** with **Add**-$6/4$ performing the best. Interestingly, we also find a value for the thresholds that outperform the configuration used in OpenCV.

6. Conclusion

Recommendations: Based on our experiences, we recommend the following. If f is *known* to be additive, the decomposition is known and d is small enough so that $\widetilde{\varphi}_t$ can be efficiently optimised, then running **Add-GP-UCB** with the known decomposition is likely to produce the best results. If not, then use a small value for d and run **Add-GP-UCB** while partially optimising for the decomposition periodically (Section 4.4). In our experiments we found that using d between 3 an 12 seemed reasonable choices. However, note that this depends on the computational budget for optimising the acquisition, the query budget for f and to a certain extent the the function f itself.



(a)

(b)

Figure 5. Results on the Astrophysical experiment (a) and the Viola and Jones dataset (b). The x-axis is the number of queries and the y-axis is the maximum value.

Summary: Our algorithm takes into account several practical considerations in real world GPB/ BO applications such as computational constraints in optimising the acquisition and the fact that we have to work with a relatively few data points since function evaluations are expensive. Our framework effectively addresses these concerns without considerably compromising on the statistical integrity of the model. We believe that this provides a promising direction to scale GPB/ BO methods to high dimensions.

Future Work: Our experiments indicate that our methods perform well beyond the scope suggested by our theory. Developing an analysis that takes into account the bias-variance and computational tradeoffs in approximating and optimising a non-additive function via an additive model is an interesting challenge. We also intend to extend this framework to discrete settings, other acquisition functions and handle more general decompositions.

Acknowledgements

We wish to thank Akshay Krishnamurthy and Andrew Gordon Wilson for the insightful discussions and Andreas Krause, Sham Kakade and Matthias Seeger for the helpful email conversations. This research is partly funded by DOE grant DESC0011114.

Our current analysis, specifically equation 14, has an error. We are working on resolving this and will post an update shortly. We would like to thank Felix Berkenkamp and Andreas Krause from ETH Zurich for pointing this out.

References

Auer, Peter. Using Confidence Bounds for Exploitation-exploration Trade-offs. *J. Mach. Learn. Res.*, 2003.

Azimi, Javad, Fern, Alan, and Fern, Xiaoli Z. Batch Bayesian Optimization via Simulation Matching. In *Advances in Neural Information Processing Systems*, 2010.

Bergstra, James S., Bardenet, Rémi, Bengio, Yoshua, and Kégl, Balázs. Algorithms for Hyper-Parameter Optimization. In *Advances in Neural Information Processing Systems*, 2011.

Bradski, Gary and Kaehler, Adrian. *Learning OpenCV*. O'Reilly Media Inc., 2008.

Brochu, Eric, Cora, Vlad M., and de Freitas, Nando. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. *CoRR*, 2010.

Bull, Adam D. Convergence Rates of Efficient Global Optimization Algorithms. *Journal of Machine Learning Research*, 2011.

Chen, Bo, Castro, Rui, and Krause, Andreas. Joint Optimization and Variable Selection of High-dimensional Gaussian Processes. In *Int'l Conference on Machine Learning*, 2012.

de Freitas, Nando. Talk on Current Challenges and Open Problems in Bayesian Optimization, 2014.

de Freitas, Nando, Smola, Alex J., and Zoghi, Masrour. Exponential Regret Bounds for Gaussian Process Bandits with Deterministic Observations. In *International Conference on Machine Learning*, 2012.

Denil, Misha, Bazzani, Loris, Larochelle, Hugo, and de Freitas, Nando. Learning Where to Attend with Deep Architectures for Image Tracking. *Neural Comput.*, 2012.

Djolonga, Josip, Krause, Andreas, and Cevher, Volkan. High-Dimensional Gaussian Process Bandits. In *Advances in Neural Information Processing Systems*, 2013.

Duvenaud, David K., Nickisch, Hannes, and Rasmussen, Carl Edward. Additive gaussian processes. In *Advances in Neural Information Processing Systems*, 2011.

Ghosal, Subhashis and Roy, Anindya. Posterior consistency of Gaussian process prior for nonparametric binary regression". *Annals of Statistics*, 2006.

Gonzalez, Javier, Longworth, Joseph, James, David, and Lawrence, Neil. Bayesian Optimization for Synthetic Gene Design. In *NIPS Workshop on Bayesian Optimization in Academia and Industry*, 2014.

Györfi, László, Kohler, Micael, Krzyzak, Adam, and Walk, Harro. *A Distribution Free Theory of Nonparametric Regression*. Springer Series in Statistics, 2002.

Hastie, T. J. and Tibshirani, R. J. *Generalized Additive Models*. London: Chapman & Hall, 1990.

Hoffman, Matthew D., Brochu, Eric, and de Freitas, Nando. Portfolio Allocation for Bayesian Optimization. In *Uncertainty in Artificial Intelligence*, 2011.

Hornby, G. S., Globus, A., Linden, D.S., and Lohn, J.D. Automated Antenna Design with Evolutionary Algorithms. *American Institute of Aeronautics and Astronautics*, 2006.

Jones, D. R., Perttunen, C. D., and Stuckman, B. E. Lipschitzian Optimization Without the Lipschitz Constant. *J. Optim. Theory Appl.*, 1993.

Jones, Donald R., Schonlau, Matthias, and Welch, William J. Efficient global optimization of expensive black-box functions. *J. of Global Optimization*, 1998.

Kandasamy, Kirthevasan, Schneider, Jeff, and Póczos, Barnabás. Bayesian Active Learning for Posterior Estimation. In *International Joint Conference on Artificial Intelligence*, 2015.

Lizotte, Daniel, Wang, Tao, Bowling, Michael, and Schuurmans, Dale. Automatic gait optimization with gaussian process regression. In *in Proc. of IJCAI*, pp. 944–949, 2007.

Ma, Yifei, Sutherland, Dougal J., Garnett, Roman, and Schneider, Jeff G. Active Pointillistic Pattern Search. In *International Conference on Artificial Intelligence and Statistics, AISTATS*, 2015.

Mahendran, Nimalan, Wang, Ziyu, Hamze, Firas, and de Freitas, Nando. Adaptive MCMC with Bayesian Optimization. In *Artificial Intelligence and Statistics*, 2012.

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

Mockus, J.B. and Mockus, L.J. Bayesian approach to global optimization and application to multiobjective and constrained problems. *Journal of Optimization Theory and Applications*, 1991.

Mockus, Jonas. Application of Bayesian approach to numerical methods of global and stochastic optimization. *Journal of Global Optimization*, 1994.

Osborne, M., Duvenaud, D., Garnett, R., Rasmussen, C., Roberts, S., and Ghahramani, Z. Active Learning of Model Evidence Using Bayesian Quadrature. In *Neural Information Processing Systems (NIPS)*, 2012.

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

Rasmussen, C.E. and Williams, C.K.I. *Gaussian Processes for Machine Learning*. Adaptative computation and machine learning series. University Press Group Limited, 2006.

Ravikumar, Pradeep, Lafferty, John, Liu, Han, and Wasserman, Larry. Sparse Additive Models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2009.

Seeger, MW., Kakade, SM., and Foster, DP. Information Consistency of Nonparametric Gaussian Process Methods. *IEEE Transactions on Information Theory*, 2008.

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

Srinivas, Niranjan, Krause, Andreas, Kakade, Sham, and Seeger, Matthias. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *International Conference on Machine Learning*, 2010.

Tegmark et al, M. Cosmological Constraints from the SDSS Luminous Red Galaxies. *Physical Review*, December 2006.

Thompson, W. R. On the Likelihood that one Unknown Probability Exceeds Another in View of the Evidence of Two Samples. *Biometrika*, 1933.

Viola, Paul A. and Jones, Michael J. Rapid Object Detection using a Boosted Cascade of Simple Features. In *Computer Vision and Pattern Recognition*, 2001.

Wang, Ziyu, Zoghi, Masrour, Hutter, Frank, Matheson, David, and de Freitas, Nando. Bayesian Optimization in High Dimensions via Random Embeddings. In *International Joint Conference on Artificial Intelligence*, 2013.

Yamins, Daniel, Tax, David, and Bergstra, James S. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. In *International Conference on Machine Learning*, 2013.

A. Some Auxiliary Material

A.1. Review of the GP-UCB Algorithm

In this subsection we present a brief summary of the **GP-UCB** algorithm in (Srinivas et al., 2010). The algorithm is given in Algorithm 3.

The following theorem gives the rate of convergence for **GP-UCB**. Note that under an additive kernel, this is the same rate as Theorem 5 which uses a different acquisition function. Note the differences in the choice of β_t.

Theorem 6. *(Modification of Theorem 2 in (Srinivas et al., 2010)) Suppose f is constructed by sampling $f^{(j)} \sim \mathcal{GP}(\mathbf{0}, \kappa^{(j)})$ for $j = 1, \dots, M$ and then adding them. Let all kernels $\kappa^{(j)}$ satisfy assumption 2 for some L, a, b. Further, we maximise the acquisition function $\widetilde{\varphi}_t$ to within $\zeta_0 t^{-1/2}$ accuracy at time step t. Pick $\delta \in (0,1)$ and choose*

$$\beta_t = 2\log\left(\frac{2t^2\pi^2}{\delta}\right) + 2D\log\left(Dt^3\right) \;\in\; \mathcal{O}\left(D\log t\right).$$

Then, **GP-UCB** *attains cumulative regret $R_T \in \mathcal{O}\left(\sqrt{D\gamma_T T\log T}\right)$ and hence simple regret $S_T \in \mathcal{O}\left(\sqrt{D\gamma_T \log T/T}\right)$. Precisely, with probability $> 1-\delta$,*

$$\forall T \geq 1, \quad R_T \leq \sqrt{8C_1\beta_T MT\gamma_t} + 2\zeta_0\sqrt{T} + C_2.$$

where $C_1 = 1/\log(1+\eta^{-2})$ and C_2 is a constant depending on a, b, D, δ, L and η.

Proof. Srinivas et al. (2010) bound the regret for exact maximisation of the **GP-UCB** acquisition φ_t. By following an analysis similar to our proof of Theorem 5 the regret can be shown to be the same for an $\zeta_0 t^{-1/2}$- optimal maximisation. □

Algorithm 3 GP-UCB

```
Input: Kernel $\kappa$, Input Space $\mathcal{X}$.
For $t = 1, 2 \dots$
  • $\mathcal{D}_0 \leftarrow \varnothing$,
  • $(\mu_0, \kappa_0) \leftarrow (\mathbf{0}, \kappa)$
  • for $t = 1, 2, \dots$
      1. $\mathbf{x}_t \leftarrow \mathrm{argmax}_{z\in\mathcal{X}}\, \mu_{t-1}(z) + \sqrt{\beta_t}\sigma_{t-1}(z)$
      2. $\mathbf{y}_t \leftarrow$ Query $f$ at $\mathbf{x}_t$.
      3. $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, \mathbf{y}_t)\}$.
      4. Perform Bayesian posterior updates to obtain $\mu_t, \sigma_t$ for $j = 1, \dots, M$.
```

A.2. Sequential Optimisation Approaches

If the function is known to be additive, we could consider several other approaches for maximisation. We list two of them here and explain their deficiencies. We recommend that the reader read the main text before reading this section.

A.2.1. Optimise one group and proceed to the next

First, fix the coordinates of $x^{(j)}, j \neq 1$ and optimise w.r.t $x^{(1)}$ by querying the function for a pre-specified number of times. Then we proceed sequentially optimising with respect to $x^{(2)}, x^{(3)} \dots$. We have outlined this algorithm in Algorithm 4. There are several reasons this approach is not desirable.

- First, it places too much faith on the additive assumption and requires that we know the decomposition at the start of the algorithm. Note that this strategy will only have searched the space in M d-dimensional subspaces. In our approach even if the function is not additive we can still hope to do well since we learn the best additive approximation to the true function. Further, if the decomposition is not known we could learn the decomposition "on the go" or at least find a reasonably good decomposition as we have explained in Section 4.4.

- Such a sequential approach is *not* an anytime algorithm. This in particular means that we need to predetermine the number of queries to be allocated to each group. After we proceed to a new group it is not straightforward to come back and improve on the solution obtained for an older group.
- This approach is not suitable for the bandits setting. We suffer large instantaneous regret up until we get to the last group. Further, after we proceed beyond a group since we cannot come back, we cannot improve on the best regret obtained in that group.

Our approach does not have any of these deficiencies.

```
Algorithm 4  Seq-Add-GP-UCB
Input: Kernels $\kappa^{(1)}, \dots, \kappa^{(M)}$, Decomposition $(\mathcal{X}^{(j)})_{j=1}^{M}$, Query Budget $T$,
  • $\mathbb{R}^D \ni \theta = \bigcup_{j=1}^{M} \theta^{(j)} = \text{rand}([0,1]^d)$
  • for $j = 1, \dots, M$
      1. $\mathcal{D}_0^{(j)} \leftarrow \varnothing$,
      2. $(\mu_0^{(j)}, \kappa_0^{(j)}) \leftarrow (\mathbf{0}, \kappa^{(j)})$.
      3. for $t = 1, 2, \dots T/M$
          (a) $\mathbf{x}_t^{(j)} \leftarrow \text{argmax}_{z \in \mathcal{X}^{(j)}} \mu^{(j)}(z) + \sqrt{\beta_t}\sigma^{(j)}(z)$
          (b) $\mathbf{x}_t \leftarrow \mathbf{x}_t^{(j)} \bigcup_{k \neq j} \theta^{(k)}$.
          (c) $\mathbf{y}_t \leftarrow$ Query $f$ at $\mathbf{x}_t$.
          (d) $\mathcal{D}_t^{(j)} = \mathcal{D}_{t-1}^{(j)} \cup \{(\mathbf{x}_t^{(j)}, \mathbf{y}_t)\}$.
          (e) Perform Bayesian posterior updates to obtain $\mu_t^{(j)}, \sigma_t^{(j)}$.
      4. $\theta^{(j)} \leftarrow \mathbf{x}_{T/M}^{(j)}$
  • Return $\theta$
```

A.2.2. Only change one Group per Query

In this strategy, the approach would be very similar to **Add-GP-UCB** except that at each query we will only update one group at time. If it is the k^{th} group the query point is determined by maximising $\widetilde{\varphi}_t^{(k)}$ for $\mathbf{x}_t^{(k)}$ and for all other groups we use values from the previous rotation. After M iterations we cycle through the groups. We have outlined this in Algorithm 5.

This is a reasonable approach and does not suffer from the same deficiencies as Algorithm 4. Maximising the acquisition function will also be slightly easier $\mathcal{O}(\zeta^{-d})$ since we need to optimise only one group at a time. However, the regret for this approach would be $\mathcal{O}(M\sqrt{D\gamma_T T \log T})$ which is a factor of M worse than the regret in our method (This can be show by following an analysis similar to the one in section B.2. This is not surprising, since at each iteration you are moving in d-coordinates of the space and you have to wait M iterations before the entire point is updated.

```
Algorithm 5  Add-GP-UCB-Buggy
Input: Kernels $\kappa^{(1)}, \dots, \kappa^{(M)}$, Decomposition $(\mathcal{X}^{(j)})_{j=1}^{M}$
  • $\mathcal{D}_0 \leftarrow \varnothing$,
  • for $j = 1, \dots, M$, $(\mu_0^{(j)}, \kappa_0^{(j)}) \leftarrow (\mathbf{0}, \kappa^{(j)})$.
  • for $t = 1, 2, \dots$
      1. $k = j \mod M$
      2. $\mathbf{x}_t^{(k)} \leftarrow \text{argmax}_{z \in \mathcal{X}^{(k)}} \mu^{(k)}(z) + \sqrt{\beta_t}\sigma^{(k)}(z)$
      3. for $j \neq k$, $\mathbf{x}_t^{(j)} \leftarrow \mathbf{x}_{t-1}^{(j)}$
      4. $\mathbf{x}_t \leftarrow \bigcup_{j=1}^{M} \mathbf{x}_t^{(j)}$.
      5. $\mathbf{y}_t \leftarrow$ Query $f$ at $\mathbf{x}_t$.
      6. $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, \mathbf{y}_t)\}$.
      7. Perform Bayesian posterior updates to obtain $\mu_t^{(j)}, \sigma_t^{(j)}$ for $j = 1, \dots, M$.
```

B. Proofs of Results in Section 4.3

B.1. Bounding the Information Gain γ_T

For this we will use the following two results from Srinivas et al. (2010).

Lemma 7. *(Information Gain in GP, (Srinivas et al., 2010) Lemma 5.3) Using the basic properties of a GP, they show that*

$$I(y_A; f_A) = \frac{1}{2}\sum_{t=1}^{n} \log(1 + \eta^{-2}\sigma_{t-1}^2(x_t)).$$

where σ_{t-1}^2 is the posterior variance after observing the first $t-1$ points.

Theorem 8. *(Bound on Information Gain, (Srinivas et al., 2010) Theorem 8) Suppose that $\mathcal{X}$ is compact and κ is a kernel on d dimensions satisfying Assumption 2. Let $n_T = C_9 T^\tau \log T$ where $C_9 = 4d + 2$. For any $T_* \in \{1, \dots, \min(T, n_T)\}$, let $B_\kappa(T_*) = \sum_{s>T_*} \lambda_s$. Here $(\lambda_n)_{n\in\mathbb{N}}$ are the eigenvalues of κ w.r.t the uniform distribution over $\mathcal{X}$. Then,*

$$\gamma_T \le \inf_{\tau} \left(\frac{1/2}{1 - e^{-1}} \max_{r\in\{1,\dots,T\}} \left(T_* \log(r n_T/\eta^2) + C_9\eta^2(1 - r/T)(T^{\tau+1}B_\kappa(T_*) + 1)\log T\right) + \mathcal{O}(T^{1-\tau/d})\right).$$

B.1.1. Proof of Theorem 4-1

Proof. We will use some bounds on the eigenvalues for the simple squared exponential kernel given in (Seeger et al., 2008). It was shown that the eigenvalues $\{\lambda_s^{(i)}\}$ of $\kappa^{(i)}$ satisfied $\lambda_s^{(i)} \le c^d B^{s^{1/d_i}}$ where $B < 1$ (See Remark 9). Since the kernel is additive, and $x^{(i)} \cap x^{(j)} = \varnothing$ the eigenfunctions corresponding to $\kappa^{(i)}$ and $\kappa^{(j)}$ will be orthogonal. Hence the eigenvalues of κ will just be the union of the eigenvalues of the individual kernels – i.e. $\{\lambda_s\} = \bigcup_{j=1}^{M}\{\lambda_s^{(j)}\}$. As $B < 1$, $\lambda_s^{(i)} \le c^d B^{s^{1/d}}$. Let $T_+ = \lfloor T_*/M \rfloor$ and $\alpha = -\log B$. Then,

$$\begin{aligned}
B_\kappa(T_*) = \sum_{s>T_*} \lambda_s &\le Mc \sum_{s>T_+} B^{s^{1/d}} \\
&\le c^d M \left(B^{T_+^{1/d}} + \int_{T_+}^{\infty} \exp(-\alpha x^{1/d})\right) \mathrm{d}x \\
&\le c^d M \left(B^{T_+^{1/d}} + d\alpha^{-d}\Gamma(d, \alpha T_+^{1/d})\right) \\
&\le c^d M e^{-\alpha T_+^{1/d}} \left(1 + d! d\alpha^{-d}(\alpha T_+^{1/d})^{d-1}\right).
\end{aligned}$$

The last step holds true whenever $\alpha T_+^{1/d} \ge 1$. Here in the second step we bound the series by an integral and in the third step we used the substitution $y = \alpha x^{1/d}$ to simplify the integral. Here $\Gamma(s, x) = \int_x^\infty t^{s-1}e^{-t}\mathrm{d}t$ is the (upper) incomplete Gamma function. In the last step we have used the following identity and the bound for integral s and $x \ge 1$

$$\Gamma(s, x) = (s-1)! e^{-x} \sum_{k=0}^{s-1} \frac{x^k}{k!} \le s! e^{-x} x^{d-1}.$$

By using $\tau = d$ and by using $T_* \le (M+1)T_+$, we use Theorem 8 to obtain the following bound on γ_T,

$$\begin{aligned}
\gamma_T \le \frac{1/2}{1 - e^{-1}} \max_{r\in\{1,\dots,T\}} &\Bigg((M+1)T_+ \log(r n_T/\eta^2) + \\
&C_9\eta^2(1 - r/T)\log T \left(1 + c^d M e^{-\alpha T_+^{1/d}} T^{d+1}\left(1 + d! d\alpha^{-d}(\alpha T_+^{1/d})^{d-1}\right)\right)\Bigg).
\end{aligned} \tag{8}$$

Now we need to pick T_+ so as to balance these two terms. We will choose $T_+ = \left(\frac{\log(T n_T)}{\alpha}\right)^d$ which is less than $\min(T, n_T)/M$ for sufficiently large T. Then $e^{-\alpha T_+^{1/d}} = 1/T n_T$. Then the first term S_1 inside the paranthesis is,

$$S_1 = (M+1)\log^d\left(\frac{T n_T}{\alpha}\right)\log\left(\frac{r n_T}{\eta^2}\right) \in \mathcal{O}\left(M\left(\log(T n_T)\right)^d \log(r n_T)\right)$$

$$
\begin{aligned}
&\in \mathcal{O}\left(M\left(\log(T^{d+1}\log T)\right)^{d}\log(rT^{d}\log T)\right) \\
&\in \mathcal{O}\left(Md^{d+1}(\log T)^{d+1} + Md^{d}(\log T)^{d}\log(r)\right).
\end{aligned}
$$

Note that the constant in front has exponential dependence on d but we ignore it since we already have d^d, $(\log T)^d$ terms. The second term S_2 becomes,

$$
\begin{aligned}
S_2 &= C_9\eta^2(1-r/T)\log T\left(1 + \frac{c^d M}{Tn_T}T^{d+1}\left(1 + d!d\alpha^{-d}(\log(Tn_T)^{d-1})\right)\right) \\
&\le C_9\eta^2(1-r/T)\left(\log T + \frac{c^d M}{C_9}\left(1 + d!d\alpha^{-d}(\log(Tn_T)^{d-1})\right)\right) \\
&\le C_9\eta^2(1-r/T)\left(\mathcal{O}(\log T) + \mathcal{O}(1) + \mathcal{O}(d!d^d(\log T)^{d-1})\right) \\
&\in \mathcal{O}\left((1-r/T)d!d^d(\log T)^{d-1}\right).
\end{aligned}
$$

Since S_1 dominates S_2, we should choose $r = T$ to maximise the RHS in (8). This gives us,

$$
\gamma_T \;\in\; \mathcal{O}\left(Md^{d+1}(\log T)^{d+1}\right) \;\in\; \mathcal{O}\left(Dd^{d}(\log T)^{d+1}\right).
$$

□

B.1.2. PROOF OF THEOREM 4-2

Proof. Once again, we use bounds given in (Seeger et al., 2008). It was shown that the eigenvalues $\{\lambda_s^{(i)}\}$ for $\kappa^{(i)}$ satisfied $\lambda_s^{(i)} \le c^d s^{-\frac{2\nu+d_j}{d_j}}$ (See Remark 9). By following a similar argument to above we have $\{\lambda_s\} = \bigcup_{j=1}^{M}\{\lambda_s^{(j)}\}$ and $\lambda_s^{(i)} \le c^d s^{-\frac{2\nu+d}{d}}$. Let $T_+ = \lfloor T_*/M \rfloor$. Then,

$$
B_\kappa(T_*) = \sum_{s>T_*}\lambda_s \le Mc^d\sum_{s>T_+}s^{-\frac{2\nu+d}{d}} \le Mc^d\left(T_+^{-\frac{2\nu+d}{d}} + \int_{T_+}^{\infty}s^{-\frac{2\nu+d}{d}}\right) \le C_8 2^d M T_+^{1-\frac{2\nu+d}{d}}.
$$

where C_8 is an appropriate constant. We set $T_+ = (Tn_T)^{\frac{d}{2\nu+d}}(\log(Tn_T))^{-\frac{d}{2\nu+d}}$ and accordingly we have the following bound on γ_T as a function of $T_+ \in \{1,\dots,\min(T,n_T)/M\}$,

$$
\gamma_T \le \inf_{\tau}\left(\frac{1/2}{1-e^{-1}}\max_{r\in\{1,\dots,T\}}\left((M+1)T_+\log(rn_T/\eta^2) + C_9\eta^2(1-r/T)\left(\log T + C_8 2^d M T_+\log(Tn_T)\right)\right) + \mathcal{O}(T^{1-\tau/d})\right). \tag{9}
$$

Since this is a concave function on r we can find the optimum by setting the derivative w.r.t r to be zero. We get $r \in \mathcal{O}(T/2^d\log(Tn_T))$ and hence,

$$
\begin{aligned}
\gamma_T &\in \inf_{\tau}\left(\mathcal{O}\left(MT_+\log\left(\frac{Tn_T}{2^d\log(Tn_T)}\right)\right) + \mathcal{O}\left(M2^dT_+\log(Tn_T)\right) + \mathcal{O}(T^{1-\tau/d})\right) \\
&\in \inf_{\tau}\left(\mathcal{O}\left(M2^d\log(Tn_T)\left(\frac{T^{\tau+1}\log(T)}{(\tau+1)\log(T)+\log\log T}\right)^{\frac{d}{2\nu+d}}\right) + \mathcal{O}(T^{1-\tau/d})\right) \\
&\in \inf_{\tau}\left(\mathcal{O}\left(M2^d\log(Tn_T)T^{\frac{(\tau+1)d}{2\nu+d}}\right) + \mathcal{O}(T^{1-\tau/d})\right) \\
&\in \mathcal{O}\left(M2^dT^{\frac{d(d+1)}{2\nu+d(d+1)}}\log(T)\right).
\end{aligned}
$$

Here in the second step we have substituted the values for T_+ first and then n_T. In the last step we have balanced the polynomial dependence on T in both terms by setting $\tau = \frac{2\nu d}{2\nu+d(d+1)}$.

□

Remark 9. *The eigenvalues and eigenfunctions for the kernel are defined with respect to a base distribution on $\mathcal{X}$. In the development of Theorem 8, Srinivas et al. (2010) draw n_T samples from the uniform distribution on $\mathcal{X}$. Hence, the eigenvalues/eigenfunctions should be w.r.t the uniform distribution. The bounds given in Seeger et al. (2008) are for the uniform distribution for the Matérn kernel and a Gaussian Distribution for the Squared Exponential Kernel. For the latter case, Srinivas et al. (2010) argue that the uniform distribution still satisfies the required tail constraints and therefore the bounds would only differ up to constants.*

B.2. Rates on Add-GP-UCB

Our analysis in this section draws ideas from Srinivas et al. (2010). We will try our best to stick to their same notation. However, unlike them we also handle the case where the acquisition function is optimised within some error. In the ensuing discussion, we will use $\tilde{\mathbf{x}}_t = \bigcup_j \tilde{\mathbf{x}}_t^{(j)}$ to denote the true maximiser of $\widetilde{\varphi}_t$ – i.e. $\tilde{\mathbf{x}}_t^{(j)} = \operatorname{argmax}_{z\in\mathcal{X}^{(j)}} \widetilde{\varphi}_t^{(j)}(z)$. $\mathbf{x}_t = \bigcup_j \mathbf{x}_t^{(j)}$ denotes the point chosen by **Add-GP-UCB** at the t^{th} iteration. Recall that $\mathbf{x}_t$ is $\zeta_0 t^{-1/2}$–optimal; I.e. $\widetilde{\varphi}_t(\tilde{\mathbf{x}}_t) - \widetilde{\varphi}_t(\mathbf{x}_t) \le \zeta_0 t^{-1/2}$.

Denote $p = \sum_j d_j$. π_t denotes a sequence such that $\sum_t \pi_t^{-1} = 1$. For e.g. when we use $\pi_t = \pi^2 t^2/6$ below, we obtain the rates in Theorem 5.

In what follows, we will construct discretisations $\Omega^{(j)}$ on each group $\mathcal{X}^{(j)}$ for the sake of analysis. Let $\omega_j = |\Omega^{(j)}|$ and $\omega_m = \max_j \omega_j$. The discretisation of the individual groups induces a discretisation Ω on $\mathcal{X}$ itself, $\Omega = \{\mathbf{x} = \bigcup_j \mathbf{x}^{(j)} : \mathbf{x}^{(j)} \in \Omega^{(j)}, j = 1, \dots, M\}$. Let $\omega = |\Omega| = \prod_j \omega_j$. We first establish the following two lemmas before we prove Theorem 5.

Lemma 10. *Pick $\delta \in (0,1)$ and set $\beta_t = 2\log(\omega_m M \pi_t/\delta)$. Then with probability $> 1-\delta$,*

$$\forall t \ge 1, \forall \mathbf{x} \in \Omega, \qquad |f(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \le \beta_t^{1/2} \sum_{j=1}^{M} \sigma_{t-1}^{(j)}(\mathbf{x}^{(j)}).$$

Proof. Conditioned on $\mathcal{D}_{t-1}$, at any given $\mathbf{x}$ and t we have $f(\mathbf{x}^{(j)}) \sim \mathcal{N}(\mu_{t-1}^{(j)}(\mathbf{x}^{(j)}), \sigma_{t-1}^{(j)} j)$, $\forall j = 1, \dots M$. Using the tail bound, $\mathbb{P}(z > M) \le \frac{1}{2} e^{-M^2/2}$ for $z \sim \mathcal{N}(0,1)$ we have with probability $> 1 - \delta/\omega M \pi_t$,

$$\frac{|f^{(j)}(\mathbf{x}^{(j)}) - \mu_{t-1}^{(j)}(\mathbf{x}^{(j)})|}{\sigma_{t-1}^{(j)}(\mathbf{x}^{(j)})} > \beta_t^{1/2} \le e^{-\beta_t/2} = \frac{\delta}{\omega_m M \pi_t}.$$

By using a union bound $\omega_j \le \omega_m$ times over all $\mathbf{x}^{(j)} \in \Omega^{(j)}$ and then M times over all discretisations the above holds with probability $> 1 - \delta/\pi_t$ for all $j = 1, \dots, M$ and $\mathbf{x}^{(j)} \in \Omega^{(j)}$. Therefore, we have $|f(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \le |f(\mathbf{x}^{(j)}) - \mu_{t-1}^{(j)}(\mathbf{x}^{(j)})| \le \beta_t^{1/2} \sum_j \sigma_{t-1}^{(j)}(\mathbf{x}^{(j)})$ for all $\mathbf{x} \in \Omega$. Now using the union bound on all t yields the result. □

Lemma 11. *The posterior mean μ_{t-1} for a GP whose kernel $\kappa(\cdot, x)$ is L-Lipschitz satisfies,*

$$\mathbb{P}\left(\forall t \ge 1 \quad |\mu_{t-1}(x) - \mu_{t-1}(x')| \le \left(f(\mathbf{x}_*) + \eta\sqrt{2\log(\pi_t/2\delta)}\right) L\eta^{-2} t \|x - x'\|_2\right) \ge 1 - \delta.$$

Proof. Note that for given t,

$$\mathbb{P}\left(y_t < f(\mathbf{x}_*) + \eta\sqrt{2\log(\pi_t/2\delta)}\right) \le \mathbb{P}\left(\epsilon_t/\eta < \sqrt{2\log(\pi_t/2\delta)}\right) \le \delta/\pi_t.$$

Therefore the statement is true with probability $> 1 - \delta$ for all t. Further, $\Delta \succ \eta^2 I$ implies $\|\Delta^{-1}\|_{op} \le \eta^{-2}$ and $|k(x,z) - k(x',z)| \le L\|x - x'\|$. Therefore

$$\begin{aligned} |\mu_{t-1}(x) - \mu_{t-1}(x')| &= |Y_{t-1}^\top \Delta^{-1}(k(x, X_T) - k(x', X_T)| \le \|Y_{t-1}\|_2 \|\Delta^{-1}\|_{op} \|k(x, X_{t-1}) - k(x', X_{t-1})\|_2 \\ &\le \left(f(\mathbf{x}_*) + \eta\sqrt{2\log(\pi_t/2\delta)}\right) L\eta^{-2}(t-1)\|x - x'\|_2. \end{aligned}$$

□

B.2.1. Proof of Theorem 5

Proof. First note that by Assumption 2 and the union bound we have, $\mathbb{P}(\forall i \ \sup_{x^{(j)}\in\mathcal{X}^{(j)}} |\partial f^{(j)}(x^{(j)})/\partial x_i^{(j)}| > J) \leq d_i a e^{-(J/b)^2}$. Since, $\partial f(x)/\partial x_i^{(j)} = \partial f^{(j)}(x^{(j)})/\partial x_i^{(j)}$, we have,

$$\mathbb{P}\left(\forall i = 1,\dots,D \ \sup_{x\in\mathcal{X}} \left|\frac{\partial f(x)}{\partial x_i}\right| > J\right) \leq pae^{-(J/b)^2}.$$

By setting $\delta/3 = pae^{-J^2/b^2}$ we have with probability $> 1-\delta/3$,

$$\forall x, x' \in \mathcal{X},\ |f(x) - f(x')| \leq b\sqrt{\log(3ap/\delta)}\|x - x'\|_1. \qquad (10)$$

Now, we construct a sequence of discretisations $\Omega_t^{(j)}$ satisfying $\|x^{(j)} - [x^{(j)}]_t]\|_1 \leq d_j/\tau_t \ \ \forall x^{(j)} \in \Omega_t^{(j)}$. Here, $[x^{(j)}]_t$ is the closest point to $x^{(j)}$ in $\Omega_t^{(j)}$ in an L_2 sense. A sufficient discretisation is a grid with τ_t uniformly spaced points. Then it follows that for all $x \in \Omega_t$, $\|x - [x]_t\|_1 \leq p/\tau_t$. Here Ω_t is the discretisation induced on $\mathcal{X}$ by the $\Omega_t^{(j)}$'s and $[x]_t$ is the closest point to x in Ω_t. Note that $\|x^{(j)} - [x^{(j)}]_t\|_2 \leq \sqrt{d_j}/\tau_t \ \forall x^{(j)} \in \Omega^{(j)}$ and $\|x - [x]_t\|_2 \leq \sqrt{p}/\tau_t$. We will set $\tau_t = pt^3$–therefore, $\omega_{tj} \leq (pt^3)^d \triangleq \omega_{mt}$. When combining this with (10), we get that with probability $> 1-\delta/3$, $|f(x) - f([x])| \leq b\sqrt{\log(3ap/\delta)}/t^3$. By our choice of β_t and using Lemma 10 the following is true for all $t \geq 1$ and for all $x \in \mathcal{X}$ with probability $> 1 - 2\delta/3$,

$$|f(x) - \mu_{t-1}([x]_t)| \leq |f(x) - f([x]_t)| + |f([x]_t) - \mu_{t-1}([x]_t)| \leq \frac{b\sqrt{\log(3ap/\delta)}}{t^2} + \beta_t^{1/2}\sum_{j=1}^{M}\sigma_{t-1}^{(j)}([x^{(j)}]_t). \qquad (11)$$

By Lemma 11 with probability $> 1-\delta/3$ we have,

$$\forall x \in \mathcal{X}, \quad |\mu_{t-1}(x) - \mu_{t-1}([x]_t)| \leq \frac{L\left(f(\mathbf{x}_*) + \eta\sqrt{2\log(3\pi_t/2\delta)}\right)}{\sqrt{p}\eta^2 t^2}. \qquad (12)$$

We use the above results to obtain the following bound on the instantaneous regret r_t which holds with probability $> 1-\delta$ for all $t \geq 1$,

$$\begin{aligned}
r_t &= f(\mathbf{x}_*) - f(\mathbf{x}_t) \\
&\leq \mu_{t-1}([\mathbf{x}_*]_t) + \beta_t^{1/2}\sum_{j=1}^{M}\sigma_{t-1}^{(j)}([\mathbf{x}_*^{(j)}]_t) - \mu_{t-1}([\mathbf{x}_t]_t) + \beta_t^{1/2}\sum_{j=1}^{M}\sigma_{t-1}^{(j)}([\mathbf{x}_t^{(j)}]_t) + \frac{2b\sqrt{\log(3ap/\delta)}}{t^3} \\
&\leq \frac{2b\sqrt{\log(3ap/\delta)}}{t^3} + \frac{\zeta_0}{\sqrt{t}} + \beta_t^{1/2}\left(\sum_{j=1}^{M}\sigma_{t-1}^{(j)}(\mathbf{x}_t^{(j)}) + \sum_{j=1}^{M}\sigma_{t-1}^{(j)}([\mathbf{x}_t^{(j)}]_t)\right) + \mu_{t-1}(\mathbf{x}_t) - \mu_{t-1}([\mathbf{x}_t]_t) \\
&\leq \frac{2b\sqrt{\log(3ap/\delta)}}{t^3} + \frac{L\left(f(\mathbf{x}_*) + \eta\sqrt{2\log(\pi_t/2\delta)}\right)}{\sqrt{p}\eta^2 t^2} + \frac{\zeta_0}{\sqrt{t}} + \beta_t^{1/2}\left(\sum_{j=1}^{M}\sigma_{t-1}^{(j)}(\mathbf{x}_t^{(j)}) + \sum_{j=1}^{M}\sigma_{t-1}^{(j)}([\mathbf{x}_t^{(j)}]_t)\right).
\end{aligned} \qquad (13)$$

In the first step we have applied Equation (11) at $\mathbf{x}_*$ and $\mathbf{x}_t$. In the second step we have used the fact that $\widetilde{\varphi}_t([\mathbf{x}_*]_t) \leq \widetilde{\varphi}_t(\tilde{\mathbf{x}}_t) \leq \widetilde{\varphi}_t(\mathbf{x}_t) + \zeta_0 t^{-1/2}$. In the third step we have used Equation (12).

For any $x \in \mathcal{X}$ we can bound $\sigma_t(x)^2$ as follows,

$$\sigma_t(x)^2 = \eta^2\eta^{-2}\sigma_t(x)^2 \leq \frac{1}{\log(1+\eta^{-2})}\log\left(1 + \eta^{-2}\sigma_t(x)^2\right).$$

Here we have used the fact that $u^2 \leq v^2\log(1+u^2)/\log(1+v^2)$ for $u \leq v$ and $\sigma_t(\mathbf{x})^2 \leq \kappa(x,x) = 1$. Write $C_1 = \log^{-1}(1+\eta^{-2})$. By using Jensen's inequality and Definition 3 for any set of T points $\{x_1, x_2, \dots x_T\} \subset \mathcal{X}$,

$$\left(\sum_{t=1}^{T}\sum_{j=1}^{M}\sigma_t^{(j)}(x^{(j)})\right)^2 \leq MT\sum_{t=1}^{T}\sum_{j=1}^{M}\sigma_t^{(j)}(x^{(j)})^2 \leq C_1 MT\sum_{t=1}^{T}\log\left(1+\eta^{-2}\sigma_t(x)^2\right) \leq 2C_1MT\gamma_T. \qquad (14)$$

Finally we can bound the cumulative regret with probability $> 1 - \delta$ for all $T \geq 1$ by,

$$\begin{aligned} R_T = \sum_{t=1}^{T} r_t &\leq C_2(a,b,D,L,\delta) + \zeta_0 \sum_{t=1}^{T} t^{-1/2} + \beta_T^{1/2} \left(\sum_{t=1}^{T}\sum_{j=1}^{M} \sigma_{t-1}^{(j)}(\mathbf{x}_t^{(j)}) + \sum_{t=1}^{T}\sum_{j=1}^{M} \sigma_{t-1}^{(j)}([\mathbf{x}_t^{(j)}]_t) \right) \\ &\leq C_2(a,b,D,L,\delta) + 2\zeta_0\sqrt{T} + \sqrt{8C_1\beta_T M T \gamma_T}. \end{aligned}$$

where we have used the summability of the first two terms in Equation (13). Here, for $\delta < 0.8$, the constant C_2 is given by,

$$C_2 \geq b\sqrt{\log(3ap/\delta)} + \frac{\pi^2 L f(\mathbf{x}_*)}{6\sqrt{p}\eta^2} + \frac{L\pi^{3/2}}{\sqrt{12p\delta}\eta}.$$

□