Gaussian Process Optimization with Adaptive Sketching: Scalable and No Regret

Daniele Calandriello [1] **Luigi Carratino** [2] **Alessandro Lazaric** [3] **Michal Valko** [4] **Lorenzo Rosasco** [1 2]

1. Introduction

Gaussian processes (GP) are a popular Bayesian approach for the optimization of black-box functions. Despite their effectiveness in simple problems, GP-based algorithms hardly scale to complex high-dimensional functions, as their per-iteration time and space cost is at least *quadratic* in the number of dimensions d and iterations t. Given a set of A alternative to choose from, the overall runtime $\mathcal{O}(t^3A)$ quickly becomes prohibitive. In this paper, we introduce BKB (*budgeted kernelized bandit*), an approximate GP algorithm for optimization under bandit feedback that achieves near-optimal regret (and hence near-optimal convergence rate) with near-constant per-iteration complexity and no assumption on the input space or the GP's covariance.

Combining a kernelized linear bandit algorithm (GP-UCB) with randomized matrix sketching technique (i.e., leverage score sampling), we prove that selecting inducing points based on their posterior variance gives an accurate low-rank approximation of the GP, preserving variance estimates and confidence intervals. As a consequence, BKB does not suffer from *variance starvation*, an important problem faced by many previous sparse GP approximations. Moreover, we show that our procedure selects at most $\widetilde{\mathcal{O}}(d_{\text{eff}})$ points, where d_{eff} is the *effective* dimension of the explored space, which is typically much smaller than both d and t. This greatly reduces the dimensionality of the problem, thus leading to a $\mathcal{O}(TAd_{\text{eff}}^2)$ runtime and $\mathcal{O}(Ad_{\text{eff}})$ space complexity.

2. Background

Notation. We denote by $[\mathbf{A}]_{ij}$ the (i,j) element of matrix $\mathbf{A}$, and by $\|\mathbf{x}\|_{\mathbf{A}}^2 = \mathbf{x}^\intercal \mathbf{A}\mathbf{x}$ the norm with metric $\mathbf{A}$. Finally, we denote the first T integers as $[T] := \{1, \dots, T\}$.

Online optimization under bandit feedback. Let $f : \mathcal{A} \to \mathbb{R}$ be a reward function that we wish to optimize over a set of decisions $\mathcal{A}$, also called actions or arms. For simplicity, we assume that $\mathcal{A} = \{\mathbf{x}_i\}_{i=1}^A$ is a fixed finite set of A vectors in $\mathbb{R}^d$. At each step $t \in [T]$ the learner **1)** chooses an arm $\mathbf{x}_t \in \mathcal{A}$, **2)** receives reward $y_t = f(\mathbf{x}_t) + \eta_t$, where η_t is a zero-mean noise, **3)** updates its model of the problem. The goal of the learner is to minimize its cumulative regret $R_T = \sum_{t=1}^T f(\mathbf{x}_*) - f(\mathbf{x}_t)$ w.r.t. the best arm $\mathbf{x}_*$, where $\mathbf{x}_* = \arg\max_{\mathbf{x}_i \in \mathcal{A}} f(\mathbf{x}_i)$. In particular, the objective of a *no-regret* algorithm is to have R_T/T go to zero as T grows as fast as possible. We recall that the regret is strictly related to the convergence rate and the optimization performance. In fact, let $\overline{x}_T$ be an arm chosen at random from the sequence of arms $(x_1, \dots, x_T)$ selected by the learner, then $f(\mathbf{x}_*) - \mathbb{E}[f(x_T)] \leq R_T/T$.

Gaussian process optimization and GP-UCB. A popular no-regret algorithm for optimization under bandit feedback is GP-UCB, introduced by Srinivas et al. (2010) in the context of Gaussian process optimization.

W.l.o.g. consider zero-mean $GP(0, k)$ priors and bounded covariance $k(\mathbf{x}_i, \mathbf{x}_i) \leq \kappa^2$ for all $\mathbf{x}_i \in \mathcal{A}$, and consider a set of observations $\{(\mathbf{x}_s, y_s)\}_{s=1}^t$ to have zero-mean Gaussian noise (i.e., $\eta_t \sim \mathcal{N}(0, \xi^2)$). Letting $\mathbf{X}_t = [\mathbf{x}_1, \dots, \mathbf{x}_t]^\intercal \in \mathbb{R}^{t \times d}$ be the matrix with all arms selected so far and $\mathbf{y}_t = [y_1, \dots, y_t]^\intercal$ be the corresponding observations, the posterior of the GP is still a GP and the mean and variance of the function at a test point $\mathbf{x}$ are defined as

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

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

where $\lambda = \xi^2$, $\mathbf{K}_t \in \mathbb{R}^{t \times t}$ is the matrix $[\mathbf{K}_t]_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$ constructed from all pairs $\mathbf{x}_i, \mathbf{x}_j$ in $\mathbf{X}_t$, and $\mathbf{k}_t(\mathbf{x}) = [k(\mathbf{x}_1, \mathbf{x}), \dots, k(\mathbf{x}_t, \mathbf{x})]^\intercal$.

The GP-UCB *algorithm* is a Bayesian optimization algorithm that uses a Gaussian process $GP(0, k)$ as a prior. Inspired by the optimism-in-face-of-uncertainty principle, at each time step t, GP-UCB uses the GP's posterior mean and variance to compute the score u_t of an arm $\mathbf{x}_i$

$$u_t(\mathbf{x}_i) = \mu_t(\mathbf{x}_i) + \beta_t \sigma_t(\mathbf{x}_i), \tag{3}$$

where we use the short-hand notation $\mu_t(\cdot) = \mu(\cdot \mid \mathbf{X}_t, \mathbf{y}_t)$ and $\sigma_t(\cdot) = \sigma(\cdot \mid \mathbf{X}_t)$. Finally, GP-UCB chooses the maxi-

[1]LCSL - Istituto Italiano di Tecnologia, Italy & MIT, USA [2]UNIGE - Università degli Studi di Genova, Italy [3]FAIR - Facebook AI Research Paris, France [4]INRIA Lille - Nord Europe SequeL team, France. Correspondence to: Daniele Calandriello <daniele.calandriello@iit.it>.

Proceedings of the Workshop on Negative Dependence in Machine Learning at the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

Full version appears as https://arxiv.org/abs/1903.05594.

mizer $\mathbf{x}_{t+1} = \arg\max_{\mathbf{x}_i \in \mathcal{A}} u_t(\mathbf{x}_i)$ as the next arm to evaluate. According to the score u_t, an arm $\mathbf{x}$ is likely to be selected if it has high mean reward μ_t *and* high variance σ_t, i.e., its estimated reward $\mu_t(\mathbf{x})$ is very uncertain. As a result, selecting the arm $\mathbf{x}_{t+1}$ with the largest score trades off between collecting (estimated) large reward (i.e., *exploitation*) and improving the accuracy of the posterior (i.e., *exploration*). The parameter β_t balances between these two objectives and it must be properly tuned to guarantee low regret. Srinivas et al. (2010) proposes different approaches to tune β_t for different assumptions on f and $\mathcal{A}$.

While GP-UCB is interpretable, simple to implement and provably achieves low regret, it is computationally expensive. In particular, computing $\sigma_t(\mathbf{x})$ has a complexity at least $\Omega(t^2)$ for the matrix-vector product $(\mathbf{K}_{t-1} + \xi^2\mathbf{I})^{-1}\mathbf{k}_{t-1}(\mathbf{x})$. Multiplying this for T iterations and A arms results in an overall $\mathcal{O}(AT^3)$ computational cost, which does not scale to large number of iterations T.

3. Budgeted Kernel Bandits

In this section, we introduce the BKB (*budgeted kernel bandit*) algorithm, a novel efficient approximation of GP-UCB, and we provide guarantees for its computational cost.

3.1. The algorithm

The main complexity bottleneck to compute the scores in Equation 3 is due to the fact that after t steps, the posterior GP is "supported" on *all* t previously seen arms, and as a consequence computing u_t requires computing a t dimensional vector $\mathbf{k}_t(\mathbf{x})$ and $t \times t$ matrix $\mathbf{K}_t$ respectively. To avoid this dependency we restrict both $\mathbf{k}_t$ and $\mathbf{K}_t$ to be supported on a subset $\mathcal{S}_t$ of m arms. This approach is part of the sparse Gaussian process approximation framework (Quinonero-Candela et al., 2007), or equivalently a linear bandit constrained on a subspace (Kuzborskij et al., 2019).

Approximated GP-UCB scores. Consider a subset of arm $\mathcal{S}_t = \{\mathbf{x}_i\}_{i=1}^m$ and denote by $\mathbf{X}_{\mathcal{S}_t} \in \mathbb{R}^{m\times d}$ the matrix with all arms in $\mathcal{S}_t$ as rows. Let $\mathbf{K}_{\mathcal{S}_t} \in \mathbb{R}^{m\times m}$ be the matrix constructed by evaluating the covariance k between any two pair of arms in $\mathcal{S}_t$ and $\mathbf{k}_{\mathcal{S}_t}(\mathbf{x}) = [k(\mathbf{x}_1, \mathbf{x}), \dots, k(\mathbf{x}_m, \mathbf{x})]^\intercal$. The Nyström embedding $\mathbf{z}_t(\cdot)$ associated with subset $\mathcal{S}_t$ is defined as the mapping[1] $\mathbf{z}_t(\cdot) = (\mathbf{K}_{\mathcal{S}_t}^{1/2})^+\mathbf{k}_{\mathcal{S}_t}(\cdot) : \mathbb{R}^d \to \mathbb{R}^m$, where $(\cdot)^+$ indicates the pseudo-inverse. We denote with $\mathbf{Z}_t(\mathbf{X}_t) = [\mathbf{z}_t(\mathbf{x}_1), \dots, \mathbf{z}_t(\mathbf{x}_t)]^\intercal \in \mathbb{R}^{t\times m}$ the associated matrix of points and we define $\mathbf{V}_t = \mathbf{Z}_t(\mathbf{X}_t)^\intercal\mathbf{Z}_t(\mathbf{X}_t) + \lambda\mathbf{I}$. Then, we approximate the posterior mean, variance, and UCB of the function on an arm $\mathbf{x}_i$ as

$$\begin{aligned}
\widetilde{\mu}_t(\mathbf{x}_i) =& \mathbf{z}_t(\mathbf{x}_i)^\intercal\mathbf{V}_t^{-1}\mathbf{Z}_t(\mathbf{x}_i)^\intercal\mathbf{y}_t \\
\widetilde{\sigma}_t^2(\mathbf{x}_i) =& \frac{1}{\lambda}\Big(k(\mathbf{x}_i, \mathbf{x}_i) - \mathbf{z}_t(\mathbf{x}_i)^\intercal\mathbf{Z}_t(\mathbf{X}_t)^\intercal\mathbf{Z}_t(\mathbf{X}_t)\mathbf{V}_t^{-1}\mathbf{z}_t(\mathbf{x}_i)\Big) \\
\widetilde{u}_t(\mathbf{x}_i) =& \widetilde{\mu}_t(\mathbf{x}_i) + \widetilde{\beta}_t\widetilde{\sigma}_t(\mathbf{x}_i), \qquad (4)
\end{aligned}$$

where $\widetilde{\beta}_t$ is appropriately tuned to achieve small regret in the theoretical analysis of Section 4. Finally, at each time step t, BKB selects arm $\widetilde{\mathbf{x}}_{t+1} = \arg\max_{\mathbf{x}_i \in \mathcal{A}} \widetilde{u}_t(\mathbf{x}_i)$.

Notice that in general, $\widetilde{\mu}_t$ and $\widetilde{\sigma}_t$ do not correspond to any GP posterior. Note that if we simply replace the $k(\mathbf{x}_i, \mathbf{x}_i)$ in the expression of $\widetilde{\sigma}_t^2(\mathbf{x}_i)$ by its value in the Nyström embedding, i.e., $\mathbf{z}_t(\mathbf{x}_i)^\intercal\mathbf{z}_t(\mathbf{x}_i)$, then we would recover a classical sparse GP approximations, the *subset of regressors*. Using $\mathbf{z}_t(\mathbf{x}_i)^\intercal\mathbf{z}_t(\mathbf{x}_i)$ is known to cause variance starvation, i.e., it can severely underestimate the variance of a test point $\mathbf{x}_i$ when it is far from the points in $\mathcal{S}_t$. Our formulation of $\widetilde{\sigma}_t$ is known in Bayesian literature as the *deterministic training conditional* (DTC), where it is used as a heuristic to prevent variance starvation. However, DTC does *not* correspond to a GP since it violates consistency (Quinonero-Candela et al., 2007). In this work, we justify this approach rigorously, showing that it is crucial to prove approximation guarantees necessary both for the optimization process and for the construction of the set of inducing points.

Algorithm 1 BKB

Require: Arm set $\mathcal{A}$, q, $\{\beta_t\}_{t=1}^T$
Ensure: Arm choices $\mathcal{D}_T = \{(\widetilde{\mathbf{x}}_t, y_t)\}$
  Select uniformly at random $\mathbf{x}_1$ and observe y_1
  Initialize $\mathcal{S}_1 = \{\mathbf{x}_1\}$
  for $t = \{1, \dots, T-1\}$ **do**
    Compute $\widetilde{\mu}_t(\mathbf{x}_i)$ and $\widetilde{\sigma}_t^2(\mathbf{x}_i)$ for all $\mathbf{x}_i \in \mathcal{A}$
    Select $\widetilde{\mathbf{x}}_{t+1} = \arg\max_{\mathbf{x}_i \in \mathcal{A}} \widetilde{u}_t(\mathbf{x}_i)$
    for $i = \{1, \dots, t+1\}$ **do**
      Set $\widetilde{p}_{t+1,i} = \overline{q} \cdot \widetilde{\sigma}_t^2(\widetilde{\mathbf{x}}_i)$
      Draw $q_{t+1,i} \sim Bernoulli(\widetilde{p}_{t+1,i})$
      If $q_{t+1} = 1$ include $\widetilde{\mathbf{x}}_i$ in $\mathcal{S}_{t+1}$
    end for
  end for

Choosing the inducing points. A critical aspect to effectively keep the complexity of BKB low while preserving regret guarantees is to carefully choose the inducing points to include in the subset $\mathcal{S}_t$. As the complexity of computing $\widetilde{u}_t$ scales with the size m of $\mathcal{S}_t$, a smaller set gives a faster algorithm. Conversely, the difference between $\widetilde{\mu}_t$ and $\widetilde{\sigma}_t$ and their exact counterparts depends on the accuracy of the embedding $\mathbf{z}_t$, which increases with the size of the set $\mathcal{S}_t$. Moreover, even for a fixed m, the quality of the embedding greatly depends on *which* inducing points are included. Finally, we need to take into account two important aspects of sequential optimization when choosing $\mathcal{S}_t$. First, we need to focus our approximation more on regions of $\mathcal{A}$ that are relevant to the optimization problem (i.e., high-reward arms). Second, as these regions change over time, we need to keep adapting the composition and size of $\mathcal{S}_t$ accordingly.

[1] Recall that in the exact version, $\mathbf{k}_t(\mathbf{x})$ can be seen as an embedding of any arm $\mathbf{x}$ into the space induced by all the t arms selected so far, i.e. using all selected points as inducing points.

We choose to construct $\mathcal{S}_t$ by randomly subsampling the set of arms $\widetilde{\mathbf{X}}_t$ evaluated so far, where arms are included in $\mathcal{S}_t$ with a probability proportional to their posterior variance σ_t. We report the complete BKB algorithm in Alg. 1.

We initialize $\mathcal{S}_1 = \{\widetilde{\mathbf{x}}_1\}$ by selecting an arm uniformly at random. At each step t, after selecting $\widetilde{\mathbf{x}}_{t+1}$, we must re-generate $\mathcal{S}_t$ to reflect the changes in $\widetilde{\mathbf{X}}_{t+1}$. Ideally, we would sample each arm in $\widetilde{\mathbf{X}}_{t+1}$ proportionally to σ^2_{t+1}, but this would be too computationally expensive. For efficiency we first approximate σ^2_{t+1} with σ^2_t. This is equivalent to ignoring the last arm and does not significantly impact the accuracy. We can then replace σ^2_t with $\widetilde{\sigma}^2_t$ that was already efficiently computed when constructing Equation 4. Finally, given a parameter $\overline{q} \geq 1$, we set our approximate inclusion probability as $\widetilde{p}_{t+1,i} = \overline{q}\widetilde{\sigma}^2_t(\widetilde{\mathbf{x}}_s)$. The $\overline{q}$ parameter is used to increase the inclusion probability in order to boost the overall success probability of the approximation procedure at the expense of a small increase in the size of $\mathcal{S}_{t+1}$. Given $\widetilde{p}_{t+1,i}$, we start from an empty $\mathcal{S}_{t+1}$ and iterate over all $\{\widetilde{\mathbf{x}}_i\}_{i=1}^{t+1}$, drawing $q_{t+1,i}$ from a Bernoulli distribution with probability $\widetilde{p}_{t+1,i}$. If $q_{t+1,i} = 1$, $\widetilde{\mathbf{x}}_i$ is included in $\mathcal{S}_{t+1}$.

Notice that while constructing $\mathcal{S}_t$ based on σ^2_t is a common heuristic in the sparse GP literature, it has not been yet rigorously justified. However, we can show that the GP posterior variance of an arm can be interpreted as its λ-ridge leverage score (RLS) (Alaoui & Mahoney, 2015). Leveraging existing analysis for RLS sampling and linear algebra we provide both accuracy and efficiency guarantees for this selection procedure.

3.2. Complexity analysis

Denote with $m_t = |\mathcal{S}_t|$ the size of the set $\mathcal{S}_t$ at step t. At each step, we first compute the embedding $\mathbf{z}_t(\mathbf{x}_i)$ of all arms in $\mathcal{O}(Am_t^2 + m_t^3)$ time, which corresponds to one inversion of $\mathbf{K}^{1/2}_{\mathcal{S}_t}$ and the matrix-vector product specific to each arm. We then rebuild the matrix $\mathbf{V}_t$ from scratch using all the arms observed so far, which requires $\mathcal{O}(\min\{t, A\}m_t^2)$ time. Then the inverse $\mathbf{V}_t^{-1}$ is computed in $\mathcal{O}(m_t^3)$ time. We can now efficiently compute $\widetilde{\mu}_t$, $\widetilde{\sigma}_t$, and $\widetilde{u}_t$ for all arms in $\mathcal{O}(Am_t^2)$ reusing the embeddings and $\mathbf{V}_t^{-1}$. Finally, computing all $q_{t+1,i}$ and $\mathcal{S}_{t+1}$ takes $\mathcal{O}(\min\{t+1, A\})$ using the estimated variances $\widetilde{\sigma}^2_t$. As a result, the per-step complexity is of order $\mathcal{O}\big((A + \min\{t, A\})m_T^2\big)$. Space-wise, we only need to store the embedded arms and $\mathbf{V}_t$ matrix, which takes at most $\mathcal{O}(Am_T)$ space.

Bounding the size of $\mathcal{S}_T$. We provide a bound on m_t, which directly determines the computational cost of BKB.

Theorem 1. *For a desired $0 < \varepsilon < 1$, $0 < \delta < 1$, let $\alpha = (1+\varepsilon)/(1-\varepsilon)$. If we run* BKB *with $\overline{q} \geq 6\alpha \log(4T/\delta)/\varepsilon^2$ then with probability $1-\delta$, for all $t \in [T]$ and for all $\mathbf{x} \in \mathcal{A}$*

(a) $\sigma^2_t(\mathbf{x})/\alpha \leq \widetilde{\sigma}^2_t(\mathbf{x}) \leq \alpha\sigma^2_t(\mathbf{x})$

(b) $m_t = |\mathcal{S}_t| \leq 3(1 + \kappa^2/\lambda)\alpha\overline{q}d_{\mathit{eff}}(\lambda, \widetilde{\mathbf{X}}_t)$.

Computational complexity. Combining our complexity analysis with Theorem 1 and the bound $m_T \leq \widetilde{\mathcal{O}}(d_{\text{eff}})$, we obtain a $\widetilde{\mathcal{O}}(T(A + \min\{t, A\})d^3_{\text{eff}})$ time complexity. Whenever $d_{\text{eff}} \ll T$ and $T \ll A$ this is essentially a quadratic $\mathcal{O}(T^2)$ runtime, a large improvement over the quartic $\mathcal{O}(T^4) \leq \mathcal{O}(T^3A)$ runtime of GP-UCB.

4. Regret Analysis

We are now ready to present the second main contribution of this paper, a bound on the regret achieved by BKB. To prove our result we additionally assume that the reward function f has bounded norm, i.e., $\|f\|^2_{\mathcal{H}} = \langle f, f\rangle < \infty$. We use an upper-bound $\|f\|_{\mathcal{H}} \leq F$ to properly tune $\widetilde{\beta}_t$ to the "range" of the reward. If F is not known in advance, standard guess-and-double techniques can be applied.

Theorem 2. *Assume $\|f\|_{\mathcal{H}} \leq F < \infty$. For any desired $0 < \varepsilon < 1$, $0 < \delta < 1$, $0 < \lambda$, let $\alpha = (1+\varepsilon)/(1-\varepsilon)$ and $\overline{q} \geq 6\alpha \log(4T/\delta)/\varepsilon^2$. If we run* BKB *with*

$$\widetilde{\beta}_t = 2\xi\sqrt{\alpha \log(\kappa^2 t)\left(\sum_{s=1}^{t} \widetilde{\sigma}^2_t(\mathbf{x}_s)\right) + \log(\tfrac{1}{\delta})} + \left(1 + \tfrac{1}{\sqrt{1-\varepsilon}}\right)\sqrt{\lambda}F,$$

then, with prob. $1-\delta$, BKB*'s regret R_T is bounded by*

$$R_T \leq 2(2\alpha)^{3/2}\sqrt{T}\Big(\xi d_{\mathit{eff}}(\lambda, \widetilde{\mathbf{X}}_T)\log(\kappa^2 T) + \xi\log(1/\delta) + \sqrt{\lambda F^2 d_{\mathit{eff}}(\lambda, \widetilde{\mathbf{X}}_T)\log(\kappa^2 T)}\Big).$$

Theorem 2 shows that BKB achieves exactly the same regret as (exact) GP-UCB up to small α constant and $\log(\kappa^2 T)$ multiplicative factor.[2] For instance, setting $\varepsilon = 1/2$ results in a bound only $3\log(T)$ times larger than GP-UCB. At the same time, the choice $\varepsilon = 1/2$ only accounts for a constant factor 12 in the per-step computational complexity, which is still dramatically reduced from t^2A to $d^2_{\text{eff}}A$. Moreover, it is easy to show that $d_{\text{eff}}(\lambda, \widetilde{\mathbf{X}}_T) \leq \log\det(\mathbf{K}_T/\lambda + \mathbf{I})$ so any bound on $\log\det(\mathbf{K}_T/\lambda + \mathbf{I})$ available for GP-UCB can be directly applied to BKB, e.g. up to an extra $\log(T)$ factor we also match GP-UCB's $\widetilde{\mathcal{O}}(\log(T)^{2d})$ rate for Gaussian kernel, and $\widetilde{\mathcal{O}}(d\sqrt{T})$ for linear kernel.

Another interesting aspect of BKB is that computing the trade-off parameter $\widetilde{\beta}_t$ can be done efficiently. Previous methods bounded this quantity with a loose (deterministic) upper bound (e.g., $\mathcal{O}(\log(T)^d)$ for Gaussian kernels) to avoid the large cost of computing $\log\det(\mathbf{K}_T/\lambda + \mathbf{I})$.

This regret guarantee is crucially achieved without requiring an *increasing accuracy* in our approximation. One would

[2] Here we derive a *frequentist* regret bound and thus we compare with the result of Chowdhury & Gopalan (2017) rather than the original *Bayesian* analysis of Srinivas et al. (2010).

expect that to obtain a sublinear regret the error induced by the approximation should decrease as $1/T$. Instead, in BKB the constants ε and λ that govern the accuracy level are fixed and thus it is not possible to guarantee that $\widetilde{\mu}_t$ will ever get close to μ_t everywhere. Adaptivity is key here: we can afford the same approximation level at every step because accuracy is actually increased only on a specific part of the arm set. For example, if a suboptimal arm is selected too often due to bad approximation, it will be eventually included in $\mathcal{S}_t$. After the inclusion, the approximation accuracy in the region of the suboptimal arm increases, and it would not be selected anymore. As the set of inducing points is updated *fast enough*, the impact of inaccurate approximations is limited in time, thus preventing large regret to accumulate. Note that this is a significant divergence from existing results. In particular approximation bounds that are uniformly accurate for all $\mathbf{x}_i \in \mathcal{A}$, such as those obtained with Quadrature FF (Mutny & Krause, 2018), rely on packing arguments. Due to the nature of packing, this usually causes the runtime or regret to scale exponentially in the input dimension d, and requires the kernel k to posses specific structure, e.g. to be statonary. Our new analysis avoids both of these problem.

Finally, we point out that the adaptivity of BKB allows drawing an interesting connection between learning and computational complexity. In fact, both the regret and the computation of BKB scale with the log-determinant and effective dimension of $\mathbf{K}_T$. As a result, if the problem is difficult from a learning point of view (i.e., the regret is large because of large log-determinant), then BKB automatically adapts the set $\mathcal{S}_t$ by including many more inducing points to guarantee the level of accuracy needed to solve the problem. Conversely, if the problem is simple (i.e., small regret), then BKB can greatly reduce the size of $\mathcal{S}_t$ and achieve the derived level of accuracy.

5. Discussion

As the literature in Bayesian optimization is vast and a complete review is out of the scope of this paper, we do not compare to alternative GP acquisition function, such as GP-EI or GP-PI, and we focus on approximation techniques with theoretical guarantees. Similarly, we exclude scalable variational inference based methods, even when their approximate posterior is provably accurate such as pF-DTC (Huggins et al., 2019), since they only provide guarantees for GP regression and not the harder optimization setting. We also do not discuss KERNELUCB (Valko et al., 2013), which has a tighter analysis than GP-UCB, since the algorithm construction is not efficient in practice.

Infinite arm sets. Looking at the proof of Theorem 1, the guarantees on $\widetilde{u}_t$ hold for all $\mathcal{H}$, and in Theorem 2 we only require that $\widetilde{\mathbf{x}}_{t+1} = \arg\max_{\mathbf{x}\in\mathcal{A}} \max_{\mathbf{w}\in\widetilde{C}_t} \boldsymbol{\phi}(\mathbf{x})^\intercal \mathbf{w}$ is returned. Therefore, the accuracy and regret guarantees hold also for an infinite set of arms $\mathcal{A}$. However, the search over $\mathcal{A}$ can be difficult, and in the general case maximization of a GP posterior is an NP-hard problem. We focused instead on the easier case of finite sets, where enumeration is sufficient. Note that this automatically introduces an $\Omega(A)$ runtime dependency, which could be removed if the user can provide an efficient method to solve the maximization problem on a specific infinite set $\mathcal{A}$. As an example, (Mutny & Krause, 2018) prove that a GP posterior approximated using QFF can be optimized efficiently in low dimensions. Finally, note that recomputing a new set $\mathcal{S}_t$ still requires $\min\{A, t\}d_{\text{eff}}^2$ at each step. This represent a separate bottleneck in BKB independent from the arm selection problem.

Linear bandit with matrix sketching. Our analysis is closely related to CBRAP (Yu et al., 2017) and SOFUL (Kuzborskij et al., 2019). CBRAP uses Gaussian projections to embed all arms in a lower dimensional space for efficiency. Unfortunately their approach must either use an embedded space at least $\Omega(T)$ large, which in most cases would be even slower than exact OFUL, or it incurs linear regret w.h.p. Another approach for Euclidean spaces based on matrix approximation is introduced by Kuzborskij et al. (2019). It uses Frequent Direction (Ghashami et al., 2016), a method similar to incremental PCA, to embed the arms into $\mathbb{R}^m$, where m is fixed in advance. They achieve a $\widetilde{\mathcal{O}}(TAm^2)$ runtime, and $\widetilde{\mathcal{O}}((1+\varepsilon_m)^{3/2}(d+m)\sqrt{T})$ regret, where ε_m is the sum of the $d-m$ smallest eigenvalues. However, if the tail does not decrease quickly enough this algorithm may suffer linear regret. On the same task BKB does achieve a $\widetilde{\mathcal{O}}(d\sqrt{T})$ regret, since it adaptively chooses the size of the embedding. Computationally, directly instantiating BKB to use a linear kernel would achieve a $\widetilde{\mathcal{O}}(TAm_t^2)$ runtime, matching Kuzborskij et al. (2019)'s .

Approximate GP with RFF. Traditionally, RFF approaches have been popular to transform GP optimization in a finite-dimensional problem and allow for scalability. Unfortunately GP-UCB with traditional RFF is not low-regret, as RFF are well known to suffer from variance starvation (Wang et al., 2018). Recently Mutny & Krause (2018) proposed an alternative approach based on QFF, a specialized approach to random features for stationary kernels. They achieve the same regret rate as GP-UCB and BKB, with a near-optimal $\mathcal{O}(TA\log(T)^{d+1})$ runtime, and present additional variations based on Thompson sampling and exact posterior maximization. However Quadrature based approaches apply to stationary kernel only, and require to ε-cover $\mathcal{A}$, hence they cannot escape an exponential dependency on the dimensionality d. Conversely BKB can be applied to any kernel function, and while not specifically designed for this task it also achieve a close $\widetilde{\mathcal{O}}(TA\log(T)^{3(d+1)})$ runtime. Moreover in practice the size of $\mathcal{S}_T$ can be much less than exponential in d.

References

Alaoui, A. E. and Mahoney, M. W. Fast randomized kernel methods with statistical guarantees. In *Neural Information Processing Systems*, 2015.

Chowdhury, S. R. and Gopalan, A. On kernelized multi-armed bandits. In *International Conference on Machine Learning*, pp. 844–853, 2017.

Ghashami, M., Liberty, E., Phillips, J. M., and Woodruff, D. P. Frequent directions: Simple and deterministic matrix sketching. *SIAM Journal on Computing*, 45(5):1762–1792, 2016.

Huggins, J. H., Campbell, T., Kasprzak, M., and Broderick, T. Scalable gaussian process inference with finite-data mean and variance guarantees. *International Conference on Artificial Intelligence and Statistics*, 2019.

Kuzborskij, I., Cella, L., and Cesa-Bianchi, N. Efficient linear bandits through matrix sketching. In *International Conference on Artificial Intelligence and Statistics*, 2019.

Mutny, M. and Krause, A. Efficient High Dimensional Bayesian Optimization with Additivity and Quadrature Fourier Features. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 9019–9030. Curran Associates, Inc., 2018.

Quinonero-Candela, J., Rasmussen, C. E., and Williams, C. K. Approximation methods for gaussian process regression. *Large-scale kernel machines*, pp. 203–224, 2007.

Srinivas, N., Krause, A., Seeger, M., and Kakade, S. M. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning*, pp. 1015–1022, 2010.

Valko, M., Korda, N., Munos, R., Flaounas, I., and Cristianini, N. Finite-time analysis of kernelised contextual bandits. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pp. 654–663. AUAI Press, 2013.

Wang, Z., Gehring, C., Kohli, P., and Jegelka, S. Batched large-scale bayesian optimization in high-dimensional spaces. In *International Conference on Artificial Intelligence and Statistics*, pp. 745–754, 2018.

Yu, X., Lyu, M. R., and King, I. CBRAP: Contextual bandits with random projection. In *AAAI Conference on Artificial Intelligence*, 2017.