Lifelong Bandit Optimization: No Prior and No Regret

Abstract

Machine learning algorithms are often repeatedly applied to problems with similar structure over and over again. We focus on solving a sequence of bandit optimization tasks and develop LIBO, an algorithm which *adapts* to the environment by learning from past experience and becomes more sample-efficient in the process. We assume a kernelized structure where the kernel is *unknown* but *shared* across all tasks. LIBO sequentially meta-learns a kernel that approximates the true kernel and solves the incoming tasks with the latest kernel estimate. Our algorithm can be paired with *any* kernelized or linear bandit algorithm and guarantees *oracle optimal* performance, meaning that as more tasks are solved, the regret of LIBO on each task converges to the regret of the bandit algorithm with oracle knowledge of the true kernel. Naturally, if paired with a sublinear bandit algorithm, LIBO yields a sublinear lifelong regret. We also show that direct access to the data from each task is not necessary for attaining sublinear regret. We propose F-LIBO, which solves the lifelong problem in a federated manner.

1 INTRODUCTION

A key aspect of human intelligence is our ability to harness previous experience and quickly improve when repeatedly solving similar problems. In this paper, we study how to solve a *sequence of learning problems, on related instances*, and become more efficient in the process. In particular we focus on problems which are solved through Bayesian Optimization, a.k.a. kernelized bandit algorithms (BO), where the kernel captures regularity structure of the tasks. A motivating application are AutoML systems, which perform hyper-parameter tuning for the same model on different datasets, or different models on the same dataset. We expect that the more tasks our machine learning system solves, the better the system becomes at solving the next one.

We model this as *lifelong learning*, where an agent sequentially faces kernelized bandit problems with different unknown reward functions. While prior work assumes the kernel to be known (e.g., hand-designed), we consider the kernel k^* to be *unknown*, but *shared* between the problem instances. After each bandit task, we use the previously collected data to *meta-learn a kernel function* $\hat{k}$ as a proxy for the unknown $k^\star$. We transfer knowledge across tasks by sequentially updating the meta-learned kernel and using it to solve the next task. This way, we adapt to the environment and gradually improve the bandit performance. Ideally, we would like to reach the oracle-optimal performance, i.e. the performance of a bandit algorithm with complete knowledge of the environment.

Lifelong bandit optimization is a delicate problem for two reasons. First, the success of each round of BO depends on the validity of the meta-learned kernel: We only have guaranteed convergence and sublinear regret if the reproducing kernel Hilbert space (RKHS), induced by the estimated kernel $\hat{k}$, contains the reward functions. Second, the data that is used for meta-learning is collected at the previous BO tasks. Thus, during each BO round, we not only have to quickly find reward maximizing actions, but also have to collect exploratory data that is sufficiently informative for successful meta-learning of the kernel.

We address these challenges when the true kernel is a sparse convex combination of a large number of candidate kernels. We propose an approach for meta-learning a provably consistent estimator of the true kernel, given data from previous tasks (Theorem 3.3). To ensure that this data is sufficiently informative, we interlace the queries of the BO agent with purely exploratory queries. Combining these two key ideas, we design our main algorithm, the *Lifelong Bandit Optimizer (*LIBO*)*. This algorithm is versatile since it is *agnostic to the bandit policy*, i.e. it can be wrapped around *any* kernelized or linear base

Submitted to the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023). **To be used for reviewing only**.

bandit algorithm to influence its policy and satisfy lifelong guarantees. We prove that it is *oracle-optimal*, i.e. that by using LIBO, we can eventually achieve the *same* worst-case performance as the base bandit algorithm which has oracle knowledge of the true kernel (Theorem 4.1). We *do not make assumptions* about the base bandit algorithm, and our convergence guarantees hold for many bandit solvers such as OFUL [Abbasi-Yadkori et al., 2011], GP-UCB [Srinivas et al., 2010] or GP-TS [Chowdhury and Gopalan, 2017]. Additionally, we consider a federated setting where each BO task is performed by a client node in a network and the data ought not to be exchanged with the server node due to privacy concerns. We propose the *Federated Lifelong Bandit Optimizer (*F-LIBO*)*, and show that it satisfies a guarantee similar to LIBO (Theorem 5.1). If we take GP-UCB as base bandit solver, LIBO and F-LIBO have the *same worst-case regret bound rates* as the GP-UCB solver when given *oracle knowledge* of the true kernel (Corollary 4.2 and 5.2). In Section 6 we support our theoretical findings by experiments on synthetic and real-world data in the AutoML context. Lastly, we discuss related works in Section 7.

2 PROBLEM STATEMENT

We consider a lifelong optimization setting, where an agent interacts with a sequence of black-box optimization problems, arriving one after another. Throughout the sequence of optimization tasks, the agent can adapt to the environment based on the previously collected data and improve its performance on the succeeding tasks. Formally, the agent iteratively faces bandit problems with unknown reward functions $f_1, ..., f_m$ residing in a RKHS $\mathcal{H}_{k^\star}$ that corresponds to an *unknown* kernel function $k^\star$. To impose regularity, we assume that the reward function has a bounded kernel norm $\|f\|_{k^\star} \leq B$ and that the domain $\mathcal{X} \subset \mathbb{R}^{d_0}$ is compact. The agent interacts with each task f_s for n time steps. For each task $s = 1, \dots, m$, at time step $i = 1, \dots, n$, the agent selects an action $\boldsymbol{x}_{s,i} \in \mathcal{X}$ and receives a stochastic reward via $y_{s,i} = f_s(\boldsymbol{x}_{s,i}) + \varepsilon_{s,i}$. Here, $\varepsilon_{s,i}$ are i.i.d. samples from a zero-mean sub-Gaussian noise with variance proxy σ^2. The goal of the agent is to maximize its rewards across all tasks. This can be formalized as minimizing the *lifelong regret* over m tasks of size n, defined as

$$R(m,n) := \sum_{s=1}^{m} \sum_{i=1}^{n} f_s(\boldsymbol{x}_s^\star) - f_s(\boldsymbol{x}_{s,i}),$$

where $\boldsymbol{x}_s^\star$ is a global maximum of f_s. If $\frac{R(m,n)}{mn} \to 0$ as $m, n \to \infty$ then the agent eventually converges to the global optimum of each upcoming optimization task. This property is commonly referred to as *sublinearity* of the regret. To attain a small regret, the agent maintains an estimate of the unknown reward function f_s based on its history. Typically, a kernelized regression oracle (e.g. kernel ridge regression or Gaussian Processes) is employed for this task. The choice of the kernel function plays a key role in the success and data-efficiency of the bandit optimization. If the hypothesis space $\mathcal{H}_k$ induced by the kernel k is too restrictive and does not contain the true reward functions f_s, the agent will likely never find reward maximizing actions. To prevent this, most practitioners pick a kernel with a conservatively complex kernel with a large hypothesis space that is very likely to contain $\mathcal{H}_{k^\star}$. However, the larger $\mathcal{H}_k$, the more observations it takes to form a good reward estimate, making the finding an optimal solution less efficient.

We take a data-driven approach to select the kernel. In particular, we aim to sequentially meta-learn a kernel $\hat{k}$ which approximates the true kernel $k^\star$, using the data from previous bandit tasks. Let $\mathcal{D}_s := \{(\boldsymbol{x}_{s,i}, y_{s,i})_{i \leq n}\}$ be the data corresponding to task s, and $\mathcal{D}_{1:s} := \mathcal{D}_1 \cup \dots \cup \mathcal{D}_s$ be the collection of datasets from the first s tasks. Then once the agent solves task s, we pass $\mathcal{D}_{1:s}$ to the *meta-agent*, who meta-learns a kernel $\hat{k}_s$. This kernel is then provided to the agent who uses it for solving the next BO task. Our meta-learning algorithm can be paired with any kernelized bandit algorithm and achieves sublinear lifelong regret, if the bandit algorithm achieves sublinear single-task regret given oracle knowledge of the kernel.

3 META-LEARNING KERNELS

We first present *Meta Kernelized Group Lasso (*META-KGL*)*, our approach to estimate the kernel $k^\star$, given data from previous tasks. We consider a large set of eligible known base kernels $\{k_1, \dots, k_p\}$ where $k_j : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for all $j = 1, \dots, p$ and $k_j(\boldsymbol{x}, \boldsymbol{x}') \leq 1$ for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ without loss of generality. We assume that while $k^\star$ is unknown, it is a sparse linear combination of kernels selected from this set, i.e., there exists $J^\star \subset \{1, \dots, p\}$ and $\alpha_1, \dots \alpha_p \in \mathbb{R}$ such that

$$k^\star(\boldsymbol{x}, \boldsymbol{x}') = \sum_{j \in J^\star} \alpha_j k_j(\boldsymbol{x}, \boldsymbol{x}'),$$

where $|J^\star| \ll p$. The set $\{k_1, \dots, k_p\}$ can be very large, since we prove that the cost of finding $J^\star$ depends only logarithmically on p. We further assume that each k_j corresponds to a d_j-dimensional feature map, i.e., $k_j(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}_j(\boldsymbol{x})^T \boldsymbol{\phi}_j(\boldsymbol{x}')$, where $\boldsymbol{\phi}_j \in \mathbb{R}^{d_j}$ and $d_j < \infty$. This setting generalizes the common linear bandit assumption, to also account for higher-order terms and interaction between coordinates of the input. Let $\boldsymbol{\phi}(\boldsymbol{x})$ denote the concatenated d-dimensional feature map, where $d := \sum_{j=1}^{p} d_j$ and $\boldsymbol{\phi}(\boldsymbol{x}) := [\boldsymbol{\phi}_j^T(\boldsymbol{x})]_{j \leq p}$. Then for $s = 1, \dots, m$ the reward functions can be written as $f_s(\cdot) = \sum_{j=1}^{p} \boldsymbol{\phi}_j^\top(\cdot) \boldsymbol{\beta}_s^{\star(j)}$ such that $\boldsymbol{\beta}_s^{\star(j)} = 0$ for all $j \notin J^\star$. Moreover, the RKHS norm of f_s will be equal to $\|f_s\|_{k^\star}^2 = \sum_{j=1}^{p} \|\boldsymbol{\beta}_s^{\star(j)}\|_2^2$. This kernel model is inspired by Kassraie et al. [2022], who assume $k^\star$ lies in the convex cone of the base kernels.

In the lifelong setting, we sequentially form kernel estimates $\hat{k}_s$ based on $\mathcal{D}_{1:s}$ for $s = 1, \dots, m$. In this section, we consider one snapshot of this process for $s = m$, where we

have fixed meta-training data $\mathcal{D}_{1:m}$ and meta-learn $\hat{k} := \hat{k}_m$. We assume without loss of generality (c.f. Appendix C),

$$k^\star(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{|J^\star|} \sum_{j \in J^\star} k_j(\boldsymbol{x}, \boldsymbol{x}'),$$

and minimize a sparsity inducing loss which allows us to discard kernels that do not appear in the above formulation. META-KGL first minimizes $\mathcal{L}(\boldsymbol{\beta}; \mathcal{D}_{1:m})$ over $\boldsymbol{\beta} \in \mathbb{R}^{md}$.

$$\begin{aligned}\mathcal{L}(\boldsymbol{\beta}; \mathcal{D}_{1:m}) :=& \frac{1}{|\mathcal{D}_{1:m}|} \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\beta}\|_2^2 + \lambda \sum_{j=1}^{p} \|\boldsymbol{\beta}^{(j)}\|_2 \quad (1)\\ =& \frac{1}{mn} \sum_{s=1}^{m} \sum_{i=1}^{n} \big(y_{s,i} - \sum_{j=1}^{p} \boldsymbol{\phi}_j^T(\boldsymbol{x}_{s,i}) \boldsymbol{\beta}_s{}^{(j)}\big)^2 \\ &+ \lambda \sum_{j=1}^{p} \sqrt{\sum_{s=1}^{m} \|\boldsymbol{\beta}_s{}^{(j)}\|_2^2}\end{aligned}$$

The vectorized formulation uses the following notation

$$\begin{aligned}\boldsymbol{y} &:= \Big[[y_{1,i}]_{i \le n}, \dots, [y_{m,i}]_{i \le n}\Big] \in \mathbb{R}^{mn},\\ \boldsymbol{\beta} &:= \Big[[\boldsymbol{\beta}_1{}^{(j)}]_{j \le p}, \dots, [\boldsymbol{\beta}_m{}^{(j)}]_{j \le p}\Big] \in \mathbb{R}^{md},\\ \boldsymbol{\beta}^{(j)} &:= \Big[\boldsymbol{\beta}_1{}^{(j)}, \dots, \boldsymbol{\beta}_m{}^{(j)}\Big] \in \mathbb{R}^{md_j},\\ \boldsymbol{\Phi} &:= \mathrm{diag}(\boldsymbol{\Phi}_1, \dots \boldsymbol{\Phi}_m) \in \mathbb{R}^{mn \times dm}.\end{aligned}$$

Here $\boldsymbol{\Phi}_s := (\boldsymbol{\phi}^\top(\boldsymbol{x}_{s,1}), \dots, \phi(\boldsymbol{x}_{s,n}))^T$ is the $n \times d$ feature matrix of a task s, and therefore $\boldsymbol{\Phi}$ denotes a block diagonal matrix which gathers the features across all tasks. This meta-loss function is convex, and is equivalent to the well-known Group Lasso objective [Lounici et al., 2011]. Therefore, it can be efficiently optimized using Group Lasso solvers [e.g., Massias et al., 2018] and enjoys the statistical properties of the Group Lasso, e.g., consistency and variable selection. Let $\hat{\boldsymbol{\beta}} := \arg\min \mathcal{L}(\boldsymbol{\beta}; \mathcal{D}_{1:m})$. The first term in Eq. (1) represents the squared prediction error of $\hat{\boldsymbol{\beta}}$, while the second is a regularization term that induces group sparsity in $\hat{\boldsymbol{\beta}}$. Mainly, the solutions $\hat{\boldsymbol{\beta}} = (\hat{\boldsymbol{\beta}}^{(1)}, \dots, \hat{\boldsymbol{\beta}}^{(p)})$ to this problem are group sparse, i.e. $\hat{\boldsymbol{\beta}}^{(j)} = 0$ for many of the indices $j \in \{1, \dots, p\}$. META-KGL then constructs the set of plausible kernels $\hat{J}$, by thresholding $\|\hat{\boldsymbol{\beta}}^{(j)}\|_2$ and discarding the kernels that do no appear to be influencing the data, i.e.,

$$\hat{J} := \left\{ j \mid j \in \{1, \dots, p\} \text{ s.t. } \|\hat{\boldsymbol{\beta}}^{(j)}\|_2 > \omega\sqrt{m} \right\}.$$

where $\omega > 0$ is a hyperparamter of the algorithm. We then construct the estimated kernel as

$$\hat{k}(\boldsymbol{x}, \boldsymbol{x}') := \frac{1}{|\hat{J}|} \sum_{j \in \hat{J}} k(\boldsymbol{x}, \boldsymbol{x}').$$

META-KGL is summarized in Algorithm 1. Under mild assumptions on the dataset, we can show that $\hat{k}$ converges to the true kernel $k^\star$ in probability. Our first assumption ensures that if $j \in J^\star$, i.e., k_j is active in the true kernel, then the contribution of k_j to the data is large enough to be statistically detectable under noise.

Assumption 3.1 (Beta-min)**.** *There exists $c_1 > 0$ such that for all $j \in J^\star$,*

$$\|\beta^{*(j)}\|_2 \ge c_1\sqrt{m}.$$

This assumption is commonly used in the high-dimensional statistics literature [Bühlmann and Van De Geer, 2011, Bunea et al., 2013, Zhao and Yu, 2006]. Our second assumption requires that the meta-training data is sufficiently diverse. In Proposition 4.3, we propose a policy which provably satisfies this assumption.

Assumption 3.2 (Sufficiently Informative Data)**.** *The feature matrix $\boldsymbol{\Phi} \in \mathbb{R}^{mn \times d}$ is sufficiently informative if there exists a constant $c_\kappa > 0$ such that $\kappa(\boldsymbol{\Phi}) \ge c_\kappa$ where*

$$\begin{aligned}\kappa(\boldsymbol{\Phi}) := \inf_{(J, \boldsymbol{b})} & \frac{1}{\sqrt{n}} \frac{\|\boldsymbol{\Phi}\boldsymbol{b}\|_2}{\sum_{j \in J} \|\boldsymbol{b}^{(j)}\|_2}\\ \text{s.t. } & \boldsymbol{b} \in \mathbb{R}^d \backslash \{0\}, \sum_{j \notin J} \|\boldsymbol{b}^{(j)}\|_2 \le 3 \sum_{j \in J} \|\boldsymbol{b}^{(j)}\|_2,\\ & J \subset \{1, \dots, p\}, |J| \le |J^\star|.\end{aligned}$$

Intuitively, $\kappa(\Phi)$ measures the quality of the data: data points that are almost identical decrease $\kappa(\Phi)$, and $\kappa(\Phi)$ is large when data points are diverse. If the minimum eigenvalue of $\boldsymbol{\Phi}$ is positive, Assumption 3.2 is automatically fulfilled. This type of assumption is common in the literature on representation/meta-learning for sequential decision-making [Yang et al., 2021, Cella and Pontil, 2021, Kassraie et al., 2022] and sparse linear bandits [Bastani and Bayati, 2020, Hao et al., 2020, Kim and Paik, 2019]. It is also known in the Lasso literature as the compatibility condition [Bühlmann and Van De Geer, 2011]. Given these assumptions, we show that META-KGL recovers the true kernel with high probability.

Theorem 3.3 (Consistency of META-KGL)**.** *Suppose $D_{1:m}$ satisfies Assumption 3.1 and 3.2 with constants c_1 and c_κ respectively. Set $\omega \in (0, c_1)$ and define $\bar{\omega} = \min\{\omega, c_1 - \omega\}$. Choose $\lambda = \bar{\omega} c_\kappa^2 / (8\sqrt{m})$. Then for $\sqrt{n} > 32\sigma/(\bar{\omega} c_\kappa^2)$,* META-KGL *satisfies*

$$\mathbb{P}\left[\hat{J} = J^\star\right] \ge 1 - p \exp\left(-m\left(\frac{\bar{\omega} c_\kappa^2 \sqrt{n}}{32\sigma} - 1\right)^2\right).$$

In particular, $\hat{J}$ is a consistent estimator both in n and m,

$$\lim_{n \to \infty} \mathbb{P}\left[\hat{J} = J^\star\right] = 1, \text{ and } \lim_{m \to \infty} \mathbb{P}\left[\hat{J} = J^\star\right] = 1.$$

Appendix D presents the proof to Theorem 3.3. This theorem shows that our meta-learned kernel converges to $k^\star$ as the number of meta-training tasks increases. First, this implies that the meta-learned hypothesis space includes the unknown reward functions allowing downstream bandit algorithms to provably converge to the optimum. Second, all candidate kernels k_j that are not active in $k^\star$ are eventually excluded from $\hat{k}$. By excluding all k_j with $j \in J^\star$ which are

not necessary for estimating $f_s \in \mathcal{H}_{k^\star}$, we effectively shrink the size of the hypothesis space, thereby reducing the uncertainty of the reward function estimates during bandit optimization. Compared to $k^{\text{full}} := \frac{1}{p}\sum_{j=1}^{p} k_j$, which naively uses all kernels, this leads to significant improvements in the query efficiency and performance of the bandit optimization.

Comparison with Prior Work. Kassraie et al. [2022] propose META-KEL, a Lasso-equivalent loss for meta-learning a sparse kernel, given i.i.d. offline data from i.i.d. tasks. We emphasize that is not possible to achieve lifelong guarantees by sequentially applying this algorithm. META-KGL differs from META-KEL in key points, and satisfies stronger consistency guarantees: 1) It converges to $k^\star$ as either n the number of samples per task, or m number of tasks grow. In contrast, META-KEL converges in m only. 2) META-KGL satisfies the exact recovery guarantee for $k^\star$ since $J^* = \hat{J}$ with high probability. While META-KEL only guarantees that $J^\star \subset \hat{J}$. This is not sufficient to show that meta-learning improves upon the trivial kernel choice k_{full}. Both of these properties are required in the lifelong analysis.

4 LIFELONG BANDIT OPTIMIZATION

We now use META-KGL as a building block to develop the *Lifelong Bandit Optimizer* (LIBO), an algorithm for lifelong bandit or Bayesian optimization. LIBO is paired with a BASEBO agent which can be instantiated by any kernelized bandit algorithm, e.g., GP-UCB [Srinivas et al., 2010] or GP-TS [Chowdhury and Gopalan, 2017]. For each task f_s, the BASEBO agent is given the kernel $\hat{k}_{s-1}$ meta-learned on the $s-1$ first tasks. Equipped with the kernel, BASEBO interacts with the current bandit environment, aiming to optimize its payoff by balancing exploration and exploitation.

In the lifelong setting, we not only have to explore for the sake of optimizing the current reward function f_s, but also we need to make sure to that the sequence of action-reward pairs will be sufficiently informative (in the sense of Assumption 3.2) for meta-learning $\hat{k}_s$ in the next stage. To this end, LIBO forces the base agent to select purely exploratory actions for the first n_s steps of the task, by i.i.d. sampling from uniform distribution on $\mathcal{X}$. Following Basu et al. [2021], we refer to this as *forced exploration* and use $\mathcal{D}_s^{\text{exp}} := \{(\boldsymbol{x}_{s,i}, y_{s,i}), i \le n_s\}$ to refer to the collected exploratory data of task f_s. We use a decreasing sequence $(n_1, \dots, n_m)$ as detailed below, since less exploration by BASEBO will be required once more multi-task data is collected. For steps $i > n_s$, BASEBO selects actions according to its normal bandit policy. After the agent has interacted with the current task for n steps, we pass the exploratory data $\mathcal{D}_{1:s}^{\text{exp}}$ to META-KGL to meta-learn $\hat{k}_s$. We then announce this new kernel estimate to the BASEBO agent for solving the next task $s+1$. Figure 1 visualizes this process and Algorithm 2 summarizes LIBO.

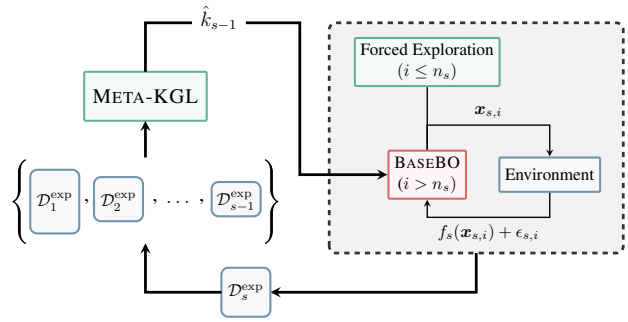

Figure 1: Overview of LIBO.

4.1 REGRET BOUNDS

Let $R^\star(n)$ be the worst-case regret of BASEBO with oracle knowledge of true kernel $k^\star$ on single tasks when the reward resides in $\mathcal{H}_{k^\star}$. When employed sequentially on m bandit tasks, the worst-case lifelong regret will be $R^\star(m,n) = mR^\star(n)$. We refer to this as oracle regret, since the BASEBO has access to the true kernel k^* which does not hold in practice. Since our meta-learned kernels $\hat{k}_s$ are an approximations of k^*, the oracle regret is a natural lower bound on the regret of LIBO.

In the following, we show that if $R^\star(n)$ the single-task oracle regret of the base bandit algorithm is sublinear (e.g., as for GP-UCB or GP-TS), then so is the lifelong regret $R(n,m)$ of LIBO. Importantly, $R(n,m)$ is not only sublinear in n, but also converges with high probability to $R^\star(m,n)$. Theorem 4.1 presents this guarantee, assuming that the forced exploration datasets $\mathcal{D}_s^{\text{exp}}$ satisfy assumption Assumption 3.2 which META-KGL requires to yield a provably consistent estimator of $k^\star$. Later in Proposition 4.3, we show that exploration by i.i.d. sampling from a uniform distribution over $\mathcal{X}$ will guarantee this assumption.

Theorem 4.1. *For all tasks $s = 1, \dots, m$, assume that the reward function $f_s \in \mathcal{H}_{k^\star}$ has bounded RKHS norm $\|f_s\|_{k^\star} \le B$. Set the number of forced exploration actions as $n_s = \frac{\sqrt{n}}{s^{1/4}}$, and assume that Assumption 3.1 and 3.2 hold for the data $\mathcal{D}_{1:s}^{\text{exp}}$ for all $s = 1, \dots, m$. Suppose, with probability greater than $1-\delta/2$,* BASEBO *has worst-case oracle regret $R^\star(m,n)$. Then, the lifelong regret of* LIBO *satisfies*

$$R(m,n) - R^\star(m,n) = \mathcal{O}\Big(\underbrace{Bm^{3/4}\sqrt{n}}_{\textit{forced exp.}} + \underbrace{B(nm)^{1/3}\log^{3/4}(mp/\delta)}_{\textit{kernel mismatch}}\Big)$$

with probability greater than $1-\delta$.

The explicit inequality without the $\mathcal{O}$-notation can be found in Appendix E, together with the proof. In the following, we give a sketch of the proof, aiming to explain the source of each term in the bound. For every forced exploration step, in the worst-case, we suffer regret of $2B$. When accumulated over a total of $\sum_{s=1}^{m} n_s$ such steps, this gives the first term in the bound. If $\hat{k}_s \neq k^\star$, it is possible to suffer from linear

regret in the worse-case. To account for this, we calculate the smallest integer m_0, for which, with high probability, $\hat{k}_s = k^\star$ for all $m_0 < s \leq m$. Based on Theorem 3.3, we show that $m_0 = \mathcal{O}((m/n^2)^{1/3} \log^{3/4}(mp/\delta))$. For every task $s \leq m_0$ we suffer a linear regret of $2Bnm_0$ in the worst-case. This is upper bounded by the second term in Theorem 4.1, which can be regarded as the cost of learning $J^\star$. Notably, it grows only logarithmically with p the number of considered features/kernels, offering a significant improvement about the polynomial rates given by prior works [e.g., Yang et al., 2021, Hong et al., 2022]. Table 2 and Table 3 present a comprehensive list of the related regret bounds.

We highlight that the excess regret of LIBO in Theorem 4.1 is sublinear in both m and n. This implies that the algorithm is *oracle optimal*, meaning that as $m \to \infty$, the single-task regret *without* knowledge of $k^\star$, eventually approaches the oracle single-task regret. Recall that $R^\star(m,n) = mR^\star(n)$ and therefore, $R(m,n)/m \to R^\star(n)$. This guarantee is stronger than that of [Basu et al., 2021, Peleg et al., 2022], where the excess regret depends linearly on m due to excessive forced exploration. By decreasing $n_s \propto s^{-1/4}$ the number of exploratory steps vanishes throughout the sequence of tasks.

As an example, we analyze the performance if GP-UCB[1] [Srinivas et al., 2010] is used as the BASEBO algorithm. In this case, we demonstrate that the worst-case lifelong regret of LIBO is of the same rate as the corresponding oracle regret. To highlight the benefit of this oracle optimality we compare to a naive baseline which uses $\hat{k}_s = k^{\text{full}} = \sum_{j=1}^p \frac{1}{p} k_j$ for all tasks instead of meta-learning $\hat{k}_s$ sequentially. In particular, we consider solving a sequence of m tasks in three scenarios: 1) running LIBO paired with GP-UCB 2) repeatedly running GP-UCB with oracle access to k^*, and 3) repeatedly running GP-UCB with k^{full}. The following corollary shows that the worst-case upper bound for the first two scenarios match in $\mathcal{O}$-notation. Appendix E.2 presents the proof.

Corollary 4.2 (Lifelong GP-UCB)**.** *Consider the setting of Theorem 4.1 with* GP-UCB *as* BASEBO *agent. Then, with probability at least* $1-\delta$*, the lifelong regret of* LIBO *paired with* GP-UCB *satisfies*

$$\begin{aligned} R(m,n) &= \mathcal{O}\big(R^\star(m,n)\big) \\ &= \mathcal{O}\left(Bmd^\star\sqrt{n}\log\tfrac{n}{d^\star} + m\sqrt{nd^\star \log\tfrac{n}{d^\star}\log\tfrac{1}{\delta}}\right) \end{aligned}$$

where $d^\star := \sum_{j\in J^\star} d_j$.

In the third scenario, we conservatively set $\hat{k}_s = k^{\text{full}}$ for all $s = 1,\dots,m$. While this is sufficient for attaining a lifelong regret that is sublinear in n, the performance will *not* be oracle optimal. In particular, this algorithm suffers from a regret of

$$R(m,n) = \mathcal{O}\left(B\tfrac{mdp}{|J^\star|}\sqrt{n}\log\tfrac{n}{d} + \sqrt{nd\log\tfrac{n}{d}\log\tfrac{1}{\delta}}\right)$$

where $d = \sum_{j=1}^p d_j \gg d^\star$ and $p/|J^\star|$ can be very large. Our experiments confirm that the performance of the naive approach is significantly worse than the other variants. This is due to the fact that confidence bounds constructed using k^{full} tend to contract slower than the ones constructed with the sparse meta-learned $\hat{k}_s$.

4.2 FORCED EXPLORATION

Our forced exploration scheme ensures that the collected data is sufficiently informative to guarantee successful meta-learning. From a technical perspective, it ensures that Assumption 3.2 is met and allows for a consistent estimator of $k^\star$. The cost of this exploration in the regret of each task is smaller in rate than the minimax regret bound [Lattimore and Szepesvári, 2020]. Therefore it has only a negligible effect on the overall performance guarantees of LIBO (see Corollary 4.2). We show that by uniformly drawing actions from the domain, the collected data satisfies this assumption:

Proposition 4.3. *Assume that* $\phi_j \in L^2(\mathcal{X})$, $j \in \{1,\dots,p\}$ *are orthonormal and let* $d_j = 1$. *Draw* $\boldsymbol{x}_1,\dots,\boldsymbol{x}_{n_1}$ *independently and uniformly from* $\mathcal{X}$*, and repeatedly use them to construct* $\mathcal{D}_{1:s}^{\text{exp}}$*. Then with probability at least* $1-\delta$*,* $\mathcal{D}_{1:s}^{\text{exp}}$ *satisfies Assumption 3.2 for* $s = 1,\dots,m$.

The proof can be found in Appendix E.3. The $d_j = 1$ condition is met without loss of generality, by splitting the higher dimensional feature maps and introducing more base features, which will increase p. Moreover, the orthonormality condition is met by orthogonalizing and re-scaling the feature maps. Basis functions such as Legendre polynomials and Fourier features [Rahimi et al., 2007] satisfy these conditions.

Generally, it is natural to require BASEBO to explore more in lifelong setting compared to when it is used in isolation and with a known kernel. We observe in our experiments that LIBO has a better empirical performance with forced exploration (i.e., $n_s > 0$) than without. This additional exploration is also required in the Representation Learning [Yang et al., 2021, 2022, Cella and Pontil, 2021, Cella et al., 2022] and hierarchical Bayesian bandit literature [Basu et al., 2021, Peleg et al., 2022, Hong et al., 2022], where it is assumed that either the context distribution or the chosen actions are diverse enough. In the case of contextual bandits, if there is sufficient randomness, the BASEBO can be greedy and yet sample diverse enough actions [Bastani et al., 2021]. Table 4 gives a detailed overview of how the related works rely on uniform exploration.

[1] Appendix E.1 provides a background on GP-UCB.

5 FEDERATED LIBO

We consider a federated extension of the lifelong learning problem. Here, each BO task is performed by a peer in a network and the corresponding data is not exchanged due to privacy concerns, limited bandwidth, etc. Operations are mainly done at the client level, and the central server only performs light computations. This setting formalizes problems such as optimizing the user experience of a software product on each user's device, e.g., for making better recommendations. Limiting the client-server communication reduces the transmit overhead time and motivates faster federated computation. Moreover, sending detailed data on user preferences and interaction patterns to the central server may jeopardize the user's privacy. However, we want to harness the statistical patterns across the user pool to improve the automated tailoring of the software product to new users. We interpret such a federated learning problem [Kairouz et al., 2021] as a client-server adaptation of our lifelong setting as described in Section 2. The meta-agent represents the server and BO tasks arise sequentially at a client node $s = 1, ..., m$ with a client specific reward function f_s.

We propose the *Federated Lifelong Bandit Optimizer* (F-LIBO) to solve this problem without directly sharing $\mathcal{D}_s$ the data corresponding to each client with the server. F-LIBO, pairs the clients and the server as follows. First, the client node s receives $\hat{k}_{s-1}$, the most recent estimate of the true kernel, and the required number of forced exploration queries n_s from the server. After taking some exploratory steps, the client performs actions according to its BASEBO policy. In contrast to LIBO, once the task is over after n steps, the client keeps $\mathcal{D}_s^{\text{exp}}$ to itself, instead of passing it back the server. The client node optimizes for a local loss

$$\begin{aligned}\hat{\boldsymbol{\beta}}_s^{(\text{client})} &:= \arg\min_{\boldsymbol{\beta}_s \in \mathbb{R}^d} \mathcal{L}\left(\boldsymbol{\beta}_s; \mathcal{D}_s^{\text{exp}}\right) \\ &= \arg\min_{\boldsymbol{\beta}_s \in \mathbb{R}^d} \frac{1}{n_s} \|\boldsymbol{y}_s - \boldsymbol{\Phi}_s \boldsymbol{\beta}_s\|_2^2 + \lambda \sum_{j=1}^{p} \|{\boldsymbol{\beta}_s}^{(j)}\|_2,\end{aligned}$$

and calculates a local estimate of $J^\star$ by thresholding $\hat{\boldsymbol{\beta}}_s^{(\text{client})}$ with the hyperparamter $\omega > 0$

$$\hat{J}_s^{(\text{client})} := \left\{ j \in \{1, \dots, p\} \text{ s.t. } \|\hat{\boldsymbol{\beta}}_s^{(\text{client})(j)}\|_2 > \omega \right\}.$$

It then sends *only* the indices $\hat{J}_s^{(\text{client})}$ back to the server. This leaves the server with the simple task of taking a α-majority vote among the s first clients, to decide which base kernels to include in $\hat{k}_s$. Formally, the server chooses

$$\hat{k}_s(\boldsymbol{x}, \boldsymbol{x}') := \frac{1}{|\hat{J}_s|} \sum_{j \in \hat{J}_s} k_j(\boldsymbol{x}, \boldsymbol{x}')$$

where for $\alpha \in [0, 1]$,

$$\hat{J}_s := \left\{ j \in \{1, \dots, p\} \text{ s.t. } \sum_{r=1}^{s} \mathbb{1}(j \in \hat{J}_r^{(\text{client})}) \geq s\alpha \right\}.$$

In other words, after client s finishes its job, the server includes the j-th kernel into its updated estimate $\hat{J}_s$, if and only if more than $s\alpha$ of the clients so far believe that it should be included. Figure 4 in the appendix visualizes this process and Algorithm 3 presents the pseudo-code to F-LIBO. Similar to LIBO, we show that if $R^\star(n)$ the worst-case oracle regret of the base bandit algorithm is sublinear in n, then so is the lifelong regret $R(n, m)$ of F-LIBO:

Theorem 5.1. *For all tasks $s = 1, \dots, m$, assume that the reward function $f_s \in \mathcal{H}_{k^\star}$ has bounded RKHS norm $\|f_s\|_{k^\star} \leq B$. Set the number of forced exploration actions as $n_s = \sqrt{n}$, and assume that Assumption 3.1 and 3.2 hold for the data $\mathcal{D}_s^{\text{exp}}$. Suppose, with probability $> 1 - \delta/2$, that* BASEBO *has worst-case oracle regret $R^\star(m, n)$. Then the lifelong regret of* F-LIBO *satisfies*

$$R(m, n) - R^\star(m, n) = \mathcal{O}\Big(Bm\sqrt{n} + B\sqrt{n}\log(mp/\delta)\Big)$$

with probability greater than $1 - \delta$.

See Appendix F.2 for the proof. Theorem 5.1 demonstrates that even without direct access to the data, the lifelong regret of F-LIBO will be sublinear in n. This theorem does not imply oracle optimality, since $R(m, n)/m - R^\star(n) \not\to 0$ for $m \to \infty$. This is due to the linear dependency of the first term on m, which arises from forced exploration. In the federated setting, we require all clients to take a fixed number of exploratory action $n_s = \sqrt{n}$, so that they have equal resources for estimating $J^\star$ and the server's majority vote is fair. We conjecture that with simple modifications, LIBO can become provably differentially private. Replacing the majority voting step with GNMax Aggregator [Papernot et al., 2018] or PRIME [Liu et al., 2021] yields a differential private voting mechanism to select $\hat{J}_s$, while preserving the lifelong regret guarantee.

Consider an example where we instantiate F-LIBO with GP-UCB as BASEBO. The worst-case regret bound of F-LIBO, which neither has knowledge of $k^\star$ nor direct access to $\mathcal{D}_s^{\text{exp}}$, matches the worst-case regret of the oracle GP-UCB in $\mathcal{O}$-notation. Corollary 5.2 formalizes this claim. Here, $R^\star(m, n)$ is the same as in Corollary 4.2.

Corollary 5.2 (Federated Lifelong GP-UCB)**.** *Consider the setting of Theorem 5.1 with* GP-UCB *as* BASEBO. *Then, with probability at least $1 - \delta$,* F-LIBO *paired with* GP-UCB *satisfies*

$$R(m, n) = \mathcal{O}\big(R^\star(m, n)\big).$$

6 EXPERIMENTS

In all experiments, we use GP-UCB as the BASEBO. We repeat all experiments with 20 random seeds and report the corresponding mean outcome with standard error. To evaluate the proposed algorithms, we use a synthetic as well as a hyper-parameter tuning environment.

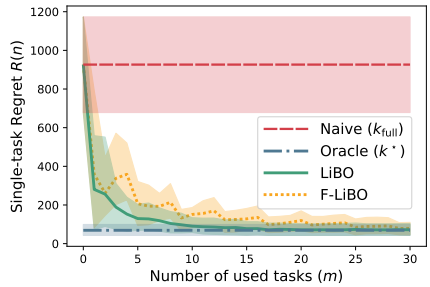

Figure 2: Singe-task cumulative regret of GP-UCB with meta-learned kernel $\hat{k}_m$ on an increasing number of meta-training tasks m.

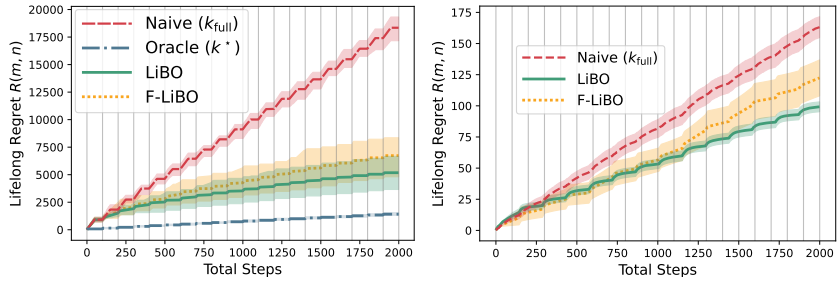

Figure 3: Lifelong cumulative regret of GP-UCB for synthetic tasks (left) and GLMNET hyperparameter tuning (right). Vertical lines indicate the beginning of a new task.

Synthetic environment. The synthetic environment is based on our data model from Section 2. We choose $\mathcal{X} = [0, 1]$ as the domain and use the first $p = 50$ cosine basis functions $\phi_j(x) = \cos(j\pi x)$ as feature maps which form the kernels $k_j(x, x') = \phi_j(x)^\top \phi_j(x')$ for $j \in \{1, ..., 50\}$. The active kernel indices $J^\star$ are sampled uniformly from the set of 5-element subsets of $\{1, ..., 50\}$, i.e., $|J^\star| = 5$. We sample the reward functions f_s independently and uniformly from $\mathcal{H}_{k^\star}$ such that $||f||_{k^\star} \leq 10$ and beta-min condition of $c_1 \geq 0.5$ holds. To the function evaluations we add i.i.d. Gaussian noise with a standard deviation of $\sigma = 0.1$.

AutoML data. A common application of Bayesian Optimization is AutoML, i.e., optimizing the hyper-parameters of machine learning algorithms. In this setting, $\mathcal{X}$ is the learning algorithm's hyper-parameter space, and f_s represents the test performance of the machine learning system. In our experiments, we consider a realistic lifelong AutoML setting where we face a sequence of hyper-parameter optimization problems. Here, each task corresponds to tuning the hyper-parameters of the GLMNET learning algorithm [Friedman et al., 2010] for a different dataset. Following previous work [e.g. Perrone et al., 2018, Rothfuss et al., 2021b], we replace the evaluation step by a table lookup based on a large number of hyper-parameter evaluations [Kühn et al., 2018] on 38 classification datasets from the OpenML platform [Bischl et al., 2017].

6.1 EXPERIMENT WITH OFFLINE DATA

We investigate how meta-leaning kernels with META-KGL and its federated variant (F-META-KGL) affects the performance of test tasks. In particular, we use meta-training data that was generated offline, based on the synthetic environment. We create data for $m = 30$ synthetic tasks, each of size $n = 10$ to be used as offline meta-data. The tasks are generated according to our synthetic environment and the details can be found in Appendix G.1. Note that $n \ll p$ i.e., we are in the overparameterized setting. We meta-learn a kernel with META-KGL and F-META-KGL using the meta-training data $\mathcal{D}_{1:s}$ for $s = 1, ..., m$, and evaluate the estimated kernel $\hat{k}$ by running GP-UCB, equipped with $\hat{k}$, for $n = 70$ iterations. Figure 2 illustrates the corresponding single-task regrets in response to increasing the number of meta-training tasks in the offline data. We report the performance of the agent that uses k^{full} (red) as a naive baseline, and the performance of an oracle agent that uses the true kernel (blue) as a natural lower bound for the achievable regret. Figure 2 shows that the regret of both meta-learned agents quickly converges to the regret of the oracle agent as the number of meta-training tasks increases. META-KGL performs slightly better than F-META-KGL, since it has direct access to all the data while the federated algorithm loses information during the voting mechanism. In Appendix G.3, we evaluate META-KGL and F-META-KGL with other choices of base kernels and higher dimensional action domains. Similar to Figure 2, we observe fast convergence to the oracle regret. Further details about the experiments are provided in Appendix G.1.

6.2 LIFELONG EXPERIMENTS

We return to the lifelong setting where the tasks arrive sequentially and evaluate our algorithms. We consider both the synthetic and AutoML environments. The horizon of each task is set to $n = 100$ time steps. To solve the synthetic problem, we consider the $p = 50$ first 1-dimensional cosine bases as the candidate feature maps. Since GLMNET has two hyper-parameters to tune, i.e., $\mathcal{X} \subset \mathbb{R}^2$, here we use the $p = 100$ first 2-dimensional cosine bases, i.e. $\phi_{i,j}(\boldsymbol{x}) = \cos(i\pi x_1)\cos(j\pi x_2)$ for $i, j = 1, ..., 10$.

Figure 3 illustrates the cumulative lifelong regrets achieved by LIBO, F-LIBO, the baseline GP-UCB with k^{full}, and oracle GP-UCB with access to $k^\star$. Note that in the AutoML environment, we do not know the true kernel $k^\star$ and thus, cannot report the oracle performance. As we would expect, LIBO and F-LIBO initially suffer the same regret as the the naive actor with k^{full} since no meta-learning data is available yet. However, as more tasks are attempted, the estimated kernel is improved and in turn, the base algorithm becomes more sample efficient on future tasks. In case of LIBO (green), over time, the forced exploration decreases and the estimated kernel converges to the true kernel. As a result, the behavior of the actor paired with the LIBO becomes indistinguishable from the actor using the oracle kernel, reflected by the same slope of the regret curves. Compared to the naive actor (red), our lifelong BO methods significantly

	oracle optimal	policy agnostic	learns sparsity	meta cost	tasks
Hu et al.	✗	✗	✗	poly d	conc
Yang et al.	✓	✗	✗	poly d	conc
Peleg et al.	✗	✗	✗	poly d	seq
Hong et al.	✓	✗	✗	poly d	seq
	✗	✗	✗	poly d	conc
LIBO	✓	✓	✓	log d	seq

Table 1: Related work (Table 4 gives a comprehensive list.)

improve the efficiency of the base agent as they accumulate more experience. In the AutoML setting, this means that we can find good hyper-parameters with fewer costly function evaluations. This showcases how incorporating knowledge transfer into deployed machine learning systems can yield significant performance gains and cost savings.

7 RELATED WORK

The lifelong bandit optimization problem addresses key shortcomings of classic kernelized bandits and Bayesian optimization. Early approaches assume that the agent knows the true kernel [Srinivas et al., 2010, Valko et al., 2013, Chowdhury and Gopalan, 2017], which is often not the case in practice. Recent work addresses this problem, either by studying the implications of misspecified kernels [Foster et al., 2020, Simchowitz et al., 2021, Bogunovic and Krause, 2021, Camilleri et al., 2021] or proposing methods for adapting kernel parameters during the optimization [Wang and de Freitas, 2014, Berkenkamp et al., 2019]. Alternatively, the appropriate kernel can be learned from related data. To this end, a number of algorithms are developed for meta-learning a kernelized Gaussian process (GP) prior [Harrison et al., 2018, Perrone et al., 2018, Rothfuss et al., 2021a,b, 2022]. However, they come without theoretical guarantees.

Theory of knowledge transfer between concurrent or sequential linear bandits has received recent attention from multiple perspectives. Representation Learning literature [Yang et al., 2021, Hu et al., 2021, Yang et al., 2022, Cella et al., 2022] assumes existence of a shared low-dimensional linear representation for the reward function, i.e. $f_s(\boldsymbol{x}) = \langle \boldsymbol{\theta}_s, \boldsymbol{B}^T \boldsymbol{x} \rangle$ where $\boldsymbol{B} \in \mathbb{R}^{d \times d^\star}$ is shared by the tasks. This matrix is unknown, however $d^\star$ is known and $d^\star \ll d$. Feature selection [Cella and Pontil, 2021] takes a similar approach by assuming that $f_s(\boldsymbol{x}) = \langle \boldsymbol{\theta}_s, \boldsymbol{S}^T \boldsymbol{x} \rangle$, where the unknown matrix $\boldsymbol{S}$ screens the relevant features $\{\boldsymbol{x}_j, j \in J^\star\}$. The elements of this matrix are 0 or 1, but contrary to representation learning, $d^\star = |J^\star|$ is unknown. Alternatively, works on Bayesian Prior learning assume existence of a shared Gaussian prior over the parameter vector, i.e. $f_s(\boldsymbol{x}) = \langle \boldsymbol{\theta}_s, \boldsymbol{x} \rangle$, where $\boldsymbol{\theta}_s \sim \mathcal{N}(\mu, \Sigma)$. This formulation does not aim for a low-dimensional solution. Following this model, Basu et al. [2021] and Hong et al. [2022] assume that Σ is known and learn distribution of μ. Peleg et al. [2022] estimate both the mean and the covariance. We consider a more relaxed setup where the mean is not shared, and meta-learn a shared covariance function. Appendix B goes into more depth to formally compare the mentioned work.

We compare LIBO with prior algorithms based on the following properties. In the context of meta-learning for BO, a desirable method is 1) oracle optimal, i.e., attains the regret guarantee of the oracle solver as m grows, 2) able to utilize any BO algorithm, 3) sample efficient, i.e., pays a small cost for meta-learning the prior/relevant features and 4) recovers low-dimensional solutions, since the effective dimension influences the sample efficiency of the base algorithm. Table 1 compares LIBO with previous work applicable to infinite action domains. Works limited to finite action sets are considered in Table 4. LIBO is the only oracle optimal algorithm that learns the effective dimension $d^\star$, while paying a cost that scales only logarithmically with the Euclidean dimension d. This is an exponential improvement compared to the polynomial dependency of prior work; moreover, it also applies to reward functions that are a linear combinations of non-linear features $f_s(\boldsymbol{x}) = \langle \boldsymbol{\theta}_s, \boldsymbol{\phi}(\boldsymbol{x}) \rangle$. Further, it can be wrapped around any linear or kernelized bandit algorithm, while earlier work require a specific bandit policy.

F-LIBO contributes to recent literature on federated learning which studies how agents can cooperate to solve a single bandit task [Dubey and Pentland, 2020, Shi et al., 2021, Huang et al., 2021, Dai et al., 2022]. In federated lifelong learning, each agent interacts with a different environment, but collaborates with others to learn relevant features.

Our work builds on ideas from Multiple-Kernel Learning [Cristianini et al., 2001, Bach et al., 2004, Ong et al., 2005, Xu et al., 2010, Gönen and Alpaydın, 2011] and Multi-Task Lasso [Obozinski et al., 2006, Argyriou et al., 2006, Lounici et al., 2011] which address consistency of model selection for offline supervised learning. Our contribution is lifelong uncertainty quantification, using a meta-learned kernel.

8 CONCLUSION

We introduce LIBO, an algorithm which allows for lifelong knowledge transfer across BO tasks trough meta-learned kernels. We show theoretically and empirically that, if paired with LIBO, the performance of a base bandit algorithm improves as more experience is gained on previous tasks. In particular, we prove that LIBO is oracle optimal in the limit. With F-LIBO, the federated variant of our main algorithm, we establish that sublinear knowledge transfer is possible even without direct access to the bandit data.

This work opens up directions of future research such as quantifying the cost of privacy in Lifelong Learning, understanding the necessity of exploration in lifelong setting, or using large neural networks to extract relevant features from prior tasks instead of working with pre-determined features.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. *NeurIPS*, 24, 2011.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *NeurIPS*, 2006.

Francis R Bach, Gert RG Lanckriet, and Michael I Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *ICML*, 2004.

Hamsa Bastani and Mohsen Bayati. Online decision making with high-dimensional covariates. *Operations Research*, 2020.

Hamsa Bastani, Mohsen Bayati, and Khashayar Khosravi. Mostly exploration-free algorithms for contextual bandits. *Management Science*, 2021.

Soumya Basu, Branislav Kveton, Manzil Zaheer, and Csaba Szepesvári. No regrets for learning the prior in bandits. *NeurIPS*, 2021.

Felix Berkenkamp, Angela P Schoellig, and Andreas Krause. No-regret bayesian optimization with unknown hyperparameters. *JMLR*, 2019.

B. Bischl, Giuseppe Casalicchio, Matthias Feurer, F. Hutter, Michel Lang, R. Mantovani, J. N. Rijn, and J. Vanschoren. Openml benchmarking suites and the openml100. *arXiv preprint*, 2017.

Ilija Bogunovic and Andreas Krause. Misspecified gaussian process bandit optimization. *NeurIPS*, 2021.

Peter Bühlmann and Sara Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

Florentina Bunea, Johannes Lederer, and Yiyuan She. The group square-root lasso: Theoretical properties and fast algorithms. *IEEE Transactions on Information Theory*, 2013.

Romain Camilleri, Kevin Jamieson, and Julian Katz-Samuels. High-dimensional experimental design and kernel bandits. In *ICML*, 2021.

Leonardo Cella and Massimiliano Pontil. Multi-task and meta-learning with sparse linear bandits. In *UAI*, 2021.

Leonardo Cella, Karim Lounici, and Massimiliano Pontil. Meta representation learning with contextual linear bandits. *arXiv preprint*, 2022.

Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. In *ICML*, 2017.

Nello Cristianini, John Shawe-Taylor, Andre Elisseeff, and Jaz Kandola. On kernel-target alignment. *NeurIPS*, 2001.

Zhongxiang Dai, Yao Shu, Arun Verma, Flint Xiaofeng Fan, Bryan Kian Hsiang Low, and Patrick Jaillet. Federated neural bandit. *arXiv preprint*, 2022.

Abhimanyu Dubey and AlexSandy Pentland. Differentially-private federated linear bandits. *NeurIPS*, 2020.

Dylan J Foster, Claudio Gentile, Mehryar Mohri, and Julian Zimmert. Adapting to misspecification in contextual bandits. In *NeurIPS*, 2020.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 2010.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *JMLR*, 2011.

Botao Hao, Tor Lattimore, and Mengdi Wang. High-dimensional sparse linear bandits. *NeurIPS*, 2020.

James Harrison, Apoorva Sharma, and Marco Pavone. Meta-learning priors for efficient online bayesian regression. In *International Workshop on the Algorithmic Foundations of Robotics*, 2018.

Joey Hong, Branislav Kveton, Manzil Zaheer, and Mohammad Ghavamzadeh. Hierarchical bayesian bandits. In *AISTATS*, 2022.

Jiachen Hu, Xiaoyu Chen, Chi Jin, Lihong Li, and Liwei Wang. Near-optimal representation learning for linear bandits and linear rl. In *ICML*, 2021.

Ruiquan Huang, Weiqiang Wu, Jing Yang, and Cong Shen. Federated linear contextual bandits. *NeurIPS*, 2021.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 2021.

Parnian Kassraie, Jonas Rothfuss, and Andreas Krause. Meta-Learning Hypothesis Spaces for Sequential Decision-making. In *ICML*, 2022.

Gi-Soo Kim and Myunghee Cho Paik. Doubly-robust lasso bandit. *NeurIPS*, 2019.

Daniel Kühn, Philipp Probst, Janek Thomas, and Bernd Bischl. Automatic exploration of machine learning experiments on openml. *arXiv preprint*, 2018.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Yingkai Li, Yining Wang, and Yuan Zhou. Nearly minimax-optimal regret for linearly parameterized bandits. In *Conference on Learning Theory*, 2019.

Yingkai Li, Yining Wang, Xi Chen, and Yuan Zhou. Tight regret bounds for infinite-armed linear contextual bandits. In *AISTATS*, 2021.

Xiyang Liu, Weihao Kong, Sham Kakade, and Sewoong Oh. Robust and differentially private mean estimation. *NeurIPS*, 2021.

Karim Lounici, Massimiliano Pontil, Sara Van De Geer, and Alexandre B Tsybakov. Oracle inequalities and optimal inference under group sparsity. *The annals of statistics*, 2011.

Mathurin Massias, Alexandre Gramfort, and Joseph Salmon. Celer: a fast solver for the lasso with dual extrapolation. In *ICML*, 2018.

Guillaume Obozinski, Ben Taskar, and Michael Jordan. Multi-task feature selection. *Statistics Department, UC Berkeley, Tech. Rep*, 2006.

Cheng Soon Ong, Alexander J. Smola, and Robert C. Williamson. Learning the kernel with hyperkernels. *JMLR*, 2005.

Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with pate. *arXiv preprint*, 2018.

Amit Peleg, Naama Pearl, and Ron Meir. Metalearning linear bandits by prior update. In *AISTATS*, 2022.

Valerio Perrone, Rodolphe Jenatton, Matthias W Seeger, and Cédric Archambeau. Scalable hyperparameter transfer learning. In *NeurIPS*, 2018.

Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *NeurIPS*, 2007.

Jonas Rothfuss, Vincent Fortuin, Martin Josifoski, and Andreas Krause. PACOH: Bayes-optimal meta-learning with PAC-guarantees. In *ICML*, 2021a.

Jonas Rothfuss, Dominique Heyn, Andreas Krause, et al. Meta-learning Reliable Priors in the Function Space. In *NeurIPS*, 2021b.

Jonas Rothfuss, Christopher Koenig, Alisa Rupenyan, and Andreas Krause. Meta-learning priors for safe bayesian optimization. In *Conference on Robot Learning (CoRL)*, 2022.

Daniel Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. *NeurIPS*, 2014.

Chengshuai Shi, Cong Shen, and Jing Yang. Federated multi-armed bandits with personalization. In *AISTATS*, 2021.

Max Simchowitz, Christopher Tosh, Akshay Krishnamurthy, Daniel J Hsu, Thodoris Lykouris, Miro Dudik, and Robert E Schapire. Bayesian decision-making under misspecified priors with applications to meta-learning. *NeurIPS*, 2021.

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *ICML*, 2010.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 1933.

Sattar Vakili, Kia Khezeli, and Victor Picheny. On information gain and regret bounds in gaussian process bandits. In *AISTATS*, 2021.

Michal Valko, Nathaniel Korda, Rémi Munos, Ilias Flaounas, and Nelo Cristianini. Finite-time analysis of kernelised contextual bandits. *arXiv preprint*, 2013.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Ziyu Wang and Nando de Freitas. Theoretical analysis of bayesian optimisation with unknown gaussian process hyper-parameters. *arXiv preprint*, 2014.

Zenglin Xu, Rong Jin, Haiqin Yang, Irwin King, and Michael R Lyu. Simple and efficient multiple kernel learning by group lasso. In *ICML*, 2010.

Jiaqi Yang, Wei Hu, Jason D. Lee, and Simon Shaolei Du. Impact of representation learning in linear bandits. In *ICLR*, 2021.

Jiaqi Yang, Qi Lei, Jason D Lee, and Simon S Du. Nearly minimax algorithms for linear bandits with shared representation. *arXiv preprint*, 2022.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *JMLR*, 2006.

A PSEUDO-CODES TO ALGORITHMS

Algorithm 1 META-KGL

Require: Data from previous tasks $\mathcal{D}_{1:s}^{\text{exp}}$, threshold parameter $\omega > 0$

$\hat{\boldsymbol{\beta}} \leftarrow \min_{\boldsymbol{\beta} \in \mathbb{R}^{sd}} \mathcal{L}\left(\boldsymbol{\beta}; \mathcal{D}_{1:s}^{\text{exp}}\right)$ ▷ solves Eq. (1)

$\hat{J} \leftarrow \{j \leq p \mid \|\hat{\boldsymbol{\beta}}^{(j)}\|_2 \geq \omega\sqrt{s}\}$

$\hat{k}_s \leftarrow \frac{1}{|\hat{J}|} \sum_{j \in \hat{J}} k_j$

Algorithm 2 LIBO

Require: $n, m \in \mathbb{N}$, $0 < \omega < c_1$, BASEBO

$\hat{k}_0 \leftarrow \sum_{j=1}^{p} \frac{1}{p} k_j$

for $s \in \{1, \ldots, m\}$ **do**

  $\mathcal{D}_s^{\text{exp}} \leftarrow \emptyset$ ▷ Dataset for kernel prediction

  $\mathcal{D}_s \leftarrow \emptyset$ ▷ Dataset for the base bandit algorithm

  for $i \in \{1, \ldots, n\}$ **do**

    if $i \leq \frac{\sqrt{n}}{s^{1/4}}$ **then** ▷ Forced exploration with rate $\sqrt{n}/s^{1/4}$

      Sample $\boldsymbol{x}_{s,i}$ uniformly from $\mathcal{X}$

      Play action $\boldsymbol{x}_{s,i}$ and observe $y_{s,i}$

      $\mathcal{D}_s^{\text{exp}} \leftarrow \mathcal{D}_s^{\text{exp}} \cup \{(\boldsymbol{x}_{s,i}, y_{s,i})\}$ ▷ Add to kernel prediction dataset

    else

      $\boldsymbol{x}_{s,i} \leftarrow$ BASEBO$(\hat{k}_{s-1})$ ▷ Select action using base bandit algorithm

      Play action $\boldsymbol{x}_{s,i}$ and observe $y_{s,i}$

    end if

    $\mathcal{D}_s \leftarrow \mathcal{D}_s \cup \{(\boldsymbol{x}_{s,i}, y_{s,i})\}$

    Update BASEBO using $\mathcal{D}_s$ ▷ Update base bandit algorithm

  end for

  $\hat{k}_s \leftarrow$ META-KGL$(\mathcal{D}_{1:s}^{\text{exp}}, \omega)$ ▷ Update $\hat{k}$ using META-KGL and $\mathcal{D}_s^{\text{exp}}$

end for

Algorithm 4 F-META-KGL

Require: $n, m \in \mathbb{N}$, data for each task $\mathcal{D}_s$, $\omega > 0$, $\alpha \in [0, 1]$

$\text{count}_1, \ldots, \text{count}_p \leftarrow 0$

for $s \in \{1, \ldots, m\}$ **do**

  $\hat{\boldsymbol{\beta}}_s \leftarrow \min_{\boldsymbol{\beta} \in \mathbb{R}^d} \mathcal{L}\left(\boldsymbol{\beta}; \mathcal{D}_s^{\text{exp}}\right)$

  for $j \in \{1, \ldots, p\}$ **do**

    $\text{count}_j \leftarrow \text{count}_j + \mathbb{1}\{\|\hat{\boldsymbol{\beta}}_s^{(j)}\|_2 \geq \omega\}$

  end for

end for

$\hat{J} \leftarrow \{j \leq p \mid \text{count}_j \geq m\alpha\}$

$\hat{k} \leftarrow \frac{1}{|\hat{J}|} \sum_{j \in \hat{J}} k_j$

B EXTENDED LITERATURE REVIEW

In this section, we present an overview of works that consider learning a potentially low-dimensional reward function by leveraging data of similar bandit tasks.

Linear Contextual Bandits with Shared Representation. The common assumption here is that the reward function for all $s \in [m]$, is linear $f_s(\boldsymbol{x}) = \langle \boldsymbol{\theta}_s, \boldsymbol{x} \rangle$ where $\boldsymbol{\theta}_s = \boldsymbol{B}\boldsymbol{w}_s$. The matrix $\boldsymbol{B} \in \mathbb{R}^{d \times r}$ is a shared representation matrix and $r \ll d$ is an intrinsic dimension. This assumption becomes more intuitive if we re-write the reward as $f_s(\boldsymbol{x}) = \langle \boldsymbol{w}_s, \boldsymbol{B}^T \boldsymbol{x} \rangle$, which implies that there exists a mapping $\boldsymbol{B} : \mathbb{R}^d \to \mathbb{R}^r$ that produces a low-dimensional representation of the actions. Our

Algorithm 3 F-LIBO

Require: $n, m \in \mathbb{N}$, $0 < \omega < c_1$, BASEBO
  $\hat{k}_0 \leftarrow \sum_{j=1}^{p} \frac{1}{p} k_j$
  for $s \in \{1, \dots, m\}$ **do**
    $\mathcal{D}_s^{\text{exp}} \leftarrow \emptyset$ ▷ Dataset for kernel prediction
    $\mathcal{D}_s \leftarrow \emptyset$ ▷ Dataset for the base bandit algorithm
    for $i \in \{1, \dots, n\}$ **do**
      if $i \leq \sqrt{n}$ **then** ▷ Forced exploration with rate $\sqrt{n}$
        Sample $\boldsymbol{x}_{s,i}$ uniformely from $\mathcal{X}$
        Play action $\boldsymbol{x}_{s,i}$ and observe $y_{s,i}$
        $\mathcal{D}_s^{\text{exp}} \leftarrow \mathcal{D}_s^{\text{exp}} \cup \{(\boldsymbol{x}_{s,i}, y_{s,i})\}$ ▷ Add to kernel prediction dataset
      else
        $\boldsymbol{x}_{s,i} \leftarrow$ BASEBO$(\hat{k}_s)$ ▷ Select action using base bandit algorithm
        Play action $\boldsymbol{x}_{s,i}$ and observe $y_{s,i}$
      end if
      $\mathcal{D}_s \leftarrow \mathcal{D}_s \cup \{(\boldsymbol{x}_{s,i}, y_{s,i})\}$
      Update BASEBO using $\mathcal{D}_s$ ▷ Update base bandit algorithm
    end for
    $\hat{k}_s \leftarrow$ F-META-KGL$(\mathcal{D}_{1:s}^{\text{exp}}, \omega)$ ▷ Update $\hat{k}$ using F-META-KGL and $\mathcal{D}_s^{\text{exp}}$
  end for

reward assumption implies that there exists a sparse matrix $\boldsymbol{S} \in \mathbb{R}^{d \times d^\star}$ which satisfies $f_s(\boldsymbol{x}) = \langle \boldsymbol{\theta}_s, \boldsymbol{S}^T \boldsymbol{\phi}(\boldsymbol{x}) \rangle$ and screens the relevant features $\boldsymbol{\phi}_j$ with $j \in J^\star$. The intrinsic dimension r then corresponds to $|J^\star|$.

Recent work on shared representation learning, often consider the contextual setting, where at every step of the bandit problem, actions may only be chosen from a set $\mathcal{A}_{s,t}$. Once the action is chosen, a noisy reward is observed. Regarding the occurrence of the tasks, two scenarios are often studied. The multi-task setting where all the tasks are solved concurrently, and the lifelong setting where the tasks arrive consecutively. Table 2 summarizes these efforts in terms of the obtained regret bounds. Here, $\tilde{\mathcal{O}}$ hides polylogarithmic factors. With the exception of [Hu et al., 2021], these works either 1) require forced exploration to fulfill sufficient exploration assumptions (SE) similar to Assumption 3.2, or 2) design a greedy algorithm assuming that the actions in set $\mathcal{A}_{s,t}$ are sampled from a diverse context distribution (DC) which gives free exploration [c.f. Bastani et al., 2021]. This suggests that for minimax optimality, either the algorithm has to explore, or the presented context should induce exploration for free. To better understand the tightness of the results in Table 2, we recall that the oracle solver which has knowledge of the representation matrix $\boldsymbol{B}$, has a lower-bound of $R^\star(n) = \Omega(\sqrt{rn \log n \log k})$, when $|\mathcal{A}_{s,t}| = k$ [Li et al., 2019]. If $\mathcal{A}_{s,t}$ is an ellipsoid, the lower-bound achievable by the oracle is $\Omega(r\sqrt{n})$ [Li et al., 2021]. Clearly, for $r \ll d$, the algorithms of Hu et al. [2021], Cella and Pontil [2021], and Cella et al. [2022] do not converge to the oracle solver as $m \to \infty$, since $R(m,n)/m \not\to R^\star(n)$.

	$\mathcal{A}_{s,t}$	r	Tasks	Expected Lifelong Regret	Base Policy	Assumptions
Yang et al. [2021]	finite	known	conc.	$\tilde{\mathcal{O}}\left(m\sqrt{rn} + \sqrt{rdnm}\right)$	Greedy	DC
	ellipsoid	known	conc.	$\tilde{\mathcal{O}}\left(mr\sqrt{n} + d^{3/2}r\sqrt{nm}\right)$	ETC	SE
Hu et al. [2021]	compact	known	conc.	$\tilde{\mathcal{O}}\left(m\sqrt{drn} + d\sqrt{rnm}\right)$	OFUL	-
Cella et al. [2022]	finite	unknown	seq.	$\tilde{\mathcal{O}}\left(mr\sqrt{n} + r\sqrt{dnm}\right)$	Greedy	DC
Cella and Pontil [2021]	finite	unknown	seq./conc.	$\tilde{\mathcal{O}}\left(mr\sqrt{n}\right)$	Greedy	DC & SE
Yang et al. [2022]	compact	known	seq.	$\tilde{\mathcal{O}}\left(mr\sqrt{n} + d\sqrt{rnm}\right)$	ETC	SE
LIBO (Ours)	compact	unknown	seq.	$\tilde{\mathcal{O}}\left(mr\sqrt{n} + m^{3/4}\sqrt{n}\right)$	any	SE

Table 2: Overview of recent work on representation learning for contextual linear bandits. Oracle lower-bound is $R^\star(m,n) = \Omega\left(mr\sqrt{n}\right)$ for infinite action set, and $R^\star(m,n) = \Omega\left(m\sqrt{rn \log n \log k}\right)$ for finite set. Polylog terms are not included.

Bayesian Bandits with Shared Prior Distributions. Alternatively, some works consider a Bayesian reward model, but without any assumption on sparsity, or low-dimensional representations. Let $f_s(\boldsymbol{x}) = \langle \boldsymbol{\theta}_s, \boldsymbol{x} \rangle$ where $\boldsymbol{\theta}_s$ are i.i.d. from $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ and the parameters $(\boldsymbol{\mu}, \Sigma)$ are shared across all tasks. Peleg et al. [2022] assume that $(\boldsymbol{\mu}, \Sigma)$ are unknown, and estimate it using the exploratory action-reward pairs collected during the first m_0 tasks. The suggested meta-algorithm can be wrapped around any Quasi-Bayesian base policy, such as Thomspon Sampling [Thompson, 1933] or Information Directed Sampling [Russo and Van Roy, 2014], however, the resulting algorithm over-explores as indicated by the $\tilde{\mathcal{O}}(md)$ term in the regret bound (See Table 3).

Taking a hierarchically Bayesian approach, Basu et al. [2021] and Hong et al. [2022] further assume that $\boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0)$ where $\boldsymbol{\mu}_0$ is unknown, but both covariance matrices Σ and Σ_0 are known. Prior distribution of $\boldsymbol{\mu}$ is updated after each task, according to the evidence collected during the task. Both papers design a meta-algorithm with Thomspon Sampling as the base solver. While Basu et al. [2021] suffers from over-exploration, Hong et al. [2022] does not require any exploration. Indeed, if Σ the covariance matrix between the actions is known, it helps with inferring rewards of other actions, and reduces the need for uniform exploration.

An overview is given in Table 3, here $R^\star(m, n)$ indicates the Bayesian lifelong regret of the oracle agent who has knowledge of $(\boldsymbol{\mu}, \Sigma)$. Note that Theorem 4.1 gives slightly stronger result, which is a high-probability bound over the regret. Here, we have taken the average to make it comparable with the Bayes regret reported in other works. As m grows, the average single-task regret is upper bounded by $R(m, n)/m$, implying that only Hong et al. [2022] and LIBO can converge to the oracle solver.

	Σ/Σ_0	sparse	Tasks	Bayesian Regret	Policy	Exp
Basu et al. [2021]	known	no	seq.	$\tilde{\mathcal{O}}\left(R^\star(m,n) + \sqrt{dnm} + md\right)$	TS	yes
Peleg et al. [2022]	unknown	no	seq.	$\tilde{\mathcal{O}}\left((1 + d^3/\sqrt{m})R^\star(m,n) + md\right)$	any QB	yes
Hong et al. [2022]	known	no	seq.	$\tilde{\mathcal{O}}\left(R^\star(m,n) + \sqrt{dmn} + d^{3/2}\right)$	TS	no
	known	no	conc.	$\tilde{\mathcal{O}}\left(R^\star(m,n) + \sqrt{dmn} + md^{3/2}\right)$	TS	yes
LIBO (Ours)	-	yes	seq.	$\tilde{\mathcal{O}}\left(R^\star(m,n) + m^{3/4}\sqrt{n} + (mn)^{1/3}\log(md)\right)$	any	yes

Table 3: Overview of recent work on meta-learning Bayesian priors for linear bandits . All works consider compact action set, except for Basu et al. [2021] which requires a finite set of actions selected from $\mathbb{R}^d$. The regret of the oracle solver is denoted by $R^\star(m, n)$.

Overall Landscape of Research. We merge the two lines of work in Table 4, to give an overview of ongoing efforts on meta-learning for linear bandits and the important properties of each method. Column $|\mathcal{A}_t|$ shows if the model holds for compact action sets, or only finite ones. Column "sparse" and "learns r" denote whether the model aims for sparse solution, and if it requires knowledge of the true sparsity r or preferably, it learns it. Column " learns Σ/Σ_0" only applies to Bayesian method, where some assume the covariance matrix of $\boldsymbol{\theta}_s$ is known, and some estimate it from data. Column "Tasks" shows if the method considers simultaneous or sequentially appearing bandit tasks. Column "O-opt" refers to oracle optimality, and has a checkmark only if $R(m, n)/m \to R^\star(n)$. Column "MS Cost" shows the cost of model selection/meta-learning. In particular, shows whether the additional regret, incurred due to not knowing the true representation/features, is logarithmic in dimension d or is it polynomial. Column "Policy" shows the base BO solvers that can be paired with the meta-learning algorithm, "any" indicates that the method can work with any (linear) policy, and "any QB" refers to Quasi-Bayesian methods. Column "Ass." shows the assumptions of the method on diversity of data, SE indicates Sufficient Exploration type assumptions, and DC refers to Diverse Context assumptions. Column "Has no uniform draws" shows if the algorithm requires forces exploration or not.

	$\lvert\mathcal{A}_t\rvert$	Sparse	Learns r	Learns Σ/Σ_0	Tasks	O-opt	MS cost	Policy	Ass.	Has no unif. draws
Yang et al. [2021]	k	✓	✗	-	conc	✓	$\text{poly}(d)$	greedy	DC	✗
	∞	✓	✗	-	conc	✓	$\text{poly}(d)$	ETC	SE	✗
Hu et al. [2021]	∞	✓	✗	-	conc	✗	$\text{poly}(d)$	OFUL	-	✓
Cella and Pontil [2021]	k	✓	✓	-	both	✗	$\log(d)$	greedy	DC SE	✗
Cella et al. [2022]	k	✓	✓	-	seq	✗	$\text{poly}(d)$	greedy	DC	✓
Yang et al. [2022]	∞	✓	✗	-	conc	✓	$\text{poly}(d)$	ETC	SE	✗
Basu et al. [2021]	k	✗	-	✗	seq	✗	$\text{poly}(d)$	TS	SE	✗
Peleg et al. [2022]	∞	✗	-	✓	seq	✗	$\text{poly}(d)$	any QB	SE	✗
Hong et al. [2022]	∞	✗	-	✗	seq	✓	$\text{poly}(d)$	TS	-	✓
	∞	✗	-	✗	conc	✗	$\text{poly}(d)$	TS	SE	✗
LIBO (Ours)	∞	✓	✓	-	seq	✓	$\log(d)$	any	SE	✗
F-LIBO (Ours)	∞	✓	✓	-	seq	✓	$\log(d)$	any	SE	✗

Table 4: Collective pro and cons of related works: LIBO gives an overall pareto-optimal solution. Refer to the corresponding paragraph in Appendix B for information on meaning of each column. Table 1 presents a concise version.

C GENERALITY OF THE KERNEL ASSUMPTION

In Section 2, we claim that, the average of kernels formulation, i.e.

$$k^\star(\cdot,\cdot) = \frac{1}{|J^\star|}\sum_{j\in J^\star} k_j(\cdot,\cdot)$$

is without loss of generality equivalent to assuming a linear combination,

$$k^\star(\cdot,\cdot) = \sum_{j\in J^\star} \alpha_j k_j(\cdot,\cdot).$$

Here, we formally show this claim. Assume there exist $\alpha_1,\dots,\alpha_p \in \mathbb{R}$ and kernels $k_1,\dots,k_p$ such that

$$k^\star(\boldsymbol{x},\boldsymbol{x}') = \sum_{j\in J^\star} \alpha_j k_j(\boldsymbol{x},\boldsymbol{x}'), \quad \forall \boldsymbol{x},\boldsymbol{x}' \in \mathcal{X}.$$

Let $f \in \mathcal{H}_{k^\star}$, then there exists $\beta_1 \in \mathbb{R}^{d_1},\dots,\beta_p \in \mathbb{R}^{d_p}$ such that for all $x \in \mathcal{X}$

$$f(\boldsymbol{x}) = \sum_{j\in J^\star} \sqrt{\alpha_j}\beta_j^\top \phi_j(\boldsymbol{x}).$$

Define $m_j := \max\{k(\boldsymbol{x},\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{X}\}$, $\tilde{\beta}_j := p m_j \beta_j \sqrt{\alpha_j}$ and $\tilde{k}_j := k_j/m_j$ for all $j \in \{1,\dots,p\}$, then

$$f(\boldsymbol{x}) = \frac{1}{p}\sum_{j\in J^\star} \tilde{\beta}_j^\top \phi_j(\boldsymbol{x})$$

and therefore $f \in \mathcal{H}_{\tilde{k}}$ for

$$\tilde{k^\star}(\boldsymbol{x},\boldsymbol{x}') = \frac{1}{|J^\star|}\sum_{j\in J^\star} k_j(\boldsymbol{x},\boldsymbol{x}').$$

This shows that the corresponding Reproducing Kernel Hilbert Spaces are equivalent, i.e. the same functions reside in both, while the norm is scaled. Therefore, we can assume, without loss of generality, that the base kernels are normalized and that the true kernel is an average of base kernels.

D CONSISTENCY OF META-KGL (PROOF OF THEOREM 3.3)

We start by proving the necessary lemmas. During this section we assume a slightly more general setting. More precisely, we assume that we have $n_s \leq n$ samples in task s, which means that the total samples size of the meta-dataset is $N = N_m := |\mathcal{D}^{\exp}| = \sum_{s=1}^{m} n_s$.

Definition D.1 (sub-Gaussian random variables)**.** *Let X be a random variable. We call X a σ* sub-Gaussian random variable *if $\mathbb{E}[X] = 0$ and*

$$\mathbb{P}[|X| \geq t] \leq 2\exp\left(\frac{t^2}{2\sigma^2}\right). \tag{D.1}$$

Lemma D.2 (Theorem 6.3.2 of Vershynin [2018])**.** *Let $\epsilon_1, \ldots, \epsilon_n$ be independent, zero mean, unit variance random variables and assume for all $i \leq n$ that ϵ_i be σ_i sub-Gaussian. Define $\epsilon = (\epsilon_1, \ldots, \epsilon_n)$ and $\sigma := \max_{i \leq n} \sigma_i$. Let $A \in \mathbb{R}^{m \times n}$ and $t \geq 0$. Then*

$$\mathbb{P}\left(\Big|\|A\epsilon\|_2 - \|A\|_F\Big| \geq t\right) \leq \exp\left(-\frac{t^2}{2\|A\|_2^2}\right).$$

Corollary D.3. *Let $\epsilon_1, \ldots, \epsilon_n$ be i.i.d. σ sub-Gaussian random variables and define $\epsilon = (\epsilon_1, \ldots, \epsilon_n)$. Let $A \in \mathbb{R}^{m \times n}$ and $t \geq \sigma\sqrt{\mathrm{tr}(AA^T)}$. Then*

$$\mathbb{P}\left(\|A\epsilon\|_2 \geq t\right) \leq \exp\left(-\frac{\left(t/\sigma - \sqrt{\mathrm{tr}(AA^T)}\right)^2}{2\|AA^T\|_2}\right).$$

Proof. The standard deviation of an σ sub-Gaussian random variable is smaller equal σ. Therefore

$$\begin{aligned}\mathbb{P}\left(\|A\epsilon\|_2 \geq t\right) &= \mathbb{P}\left(\|A\epsilon\|_2/\sqrt{Var(\epsilon_1)} \geq t/\sqrt{Var(\epsilon_1)}\right)\\ &\leq \mathbb{P}\left(\|A\epsilon\|_2/\sqrt{Var(\epsilon_1)} \geq t/\sigma\right).\end{aligned}$$

It holds that $\|A\|_F = \sqrt{\mathrm{tr}(AA^T)}$. Define $\tilde{\epsilon} = \epsilon/\sqrt{Var(\epsilon_1)}$. We have

$$\begin{aligned}\mathbb{P}\left(\|A\epsilon\|_2 \geq t\right) &\leq \mathbb{P}\left(\|A\tilde{\epsilon}\|_2 - \|A\|_F \geq t/\sigma - \sqrt{\mathrm{tr}(AA^T)}\right)\\ &\leq \mathbb{P}\left(|\|A\tilde{\epsilon}\|_2 - \|A\|_F| \geq t/\sigma - \sqrt{\mathrm{tr}(AA^T)}\right).\end{aligned}$$

Using Lemma D.2 and noting that $\|A\|_2^2 = \|AA^T\|_2$ yields the desired result. □

Lemma D.4. *Consider the model in Eq. (1) with σ sub-Gaussian i.i.d. noise. Then, for $\frac{\lambda N}{4\sigma} > \sqrt{\mathrm{tr}(\mathbf{\Phi}^{(j)}(\mathbf{\Phi}^{(j)})^T)}$ with probability at least*

$$1 - \sum_{j=1}^{p} \exp\left(-\frac{\left(\frac{\lambda N}{4\sigma} - \sqrt{\mathrm{tr}(\mathbf{\Phi}^{(j)}(\mathbf{\Phi}^{(j)})^T)}\right)^2}{2\|\mathbf{\Phi}^{(j)}(\mathbf{\Phi}^{(j)})^T\|_2}\right)$$

we have for any solution $\hat{\boldsymbol{\beta}}$ of Eq. (1)

$$\frac{1}{N}\|\mathbf{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2^2 + \frac{\lambda}{2}\sum_{j=1}^{p}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \leq 2\lambda \sum_{j \in J^\star} \min\left(\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2, \|\boldsymbol{\beta}^{\star(j)}\|_2\right).$$

Proof. This proof is inspired by the proof of Lemma 3.1 in Lounici et al. [2011].
For all solutions $\hat{\boldsymbol{\beta}}$ of Eq. (1)

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{dm}}{\arg\min} \frac{1}{N}\|\boldsymbol{\Phi}\boldsymbol{\beta}^\star + \epsilon - \boldsymbol{\Phi}\boldsymbol{\beta}\|_2^2 + \lambda\sum_{j=1}^{p}\|\boldsymbol{\beta}^{(j)}\|_2.$$

Therefore for all $\boldsymbol{\beta} \in \mathbb{R}^{dm}$

$$\frac{1}{N}\|\boldsymbol{\Phi}\boldsymbol{\beta}^\star + \epsilon - \boldsymbol{\Phi}\hat{\boldsymbol{\beta}}\|_2^2 + \lambda\sum_{j=1}^{p}\|\hat{\boldsymbol{\beta}}^{(j)}\|_2 \leq \frac{1}{N}\|\boldsymbol{\Phi}\boldsymbol{\beta}^\star + \epsilon - \boldsymbol{\Phi}\boldsymbol{\beta}\|_2^2 + \lambda\sum_{j=1}^{p}\|\boldsymbol{\beta}^{(j)}\|_2.$$

This yields

$$\begin{aligned}\frac{1}{N}\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})\|_2^2 \leq & \frac{1}{N}\|\boldsymbol{\Phi}(\boldsymbol{\beta} - \boldsymbol{\beta}^\star)\|_2^2 + \frac{2}{N}\epsilon^T\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \\ & + \lambda\sum_{j=1}^{p}\left(\|\boldsymbol{\beta}^{(j)}\|_2 - \|\hat{\boldsymbol{\beta}}^{(j)}\|_2\right).\end{aligned}$$

And in particular if $\boldsymbol{\beta} = \boldsymbol{\beta}^\star$, then,

$$\begin{aligned}\frac{1}{N}\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2^2 \leq & \frac{2}{N}\epsilon^T\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star) \\ & + \lambda\sum_{j=1}^{p}\left(\|\boldsymbol{\beta}^{\star(j)}\|_2 - \|\hat{\boldsymbol{\beta}}^{(j)}\|_2\right).\end{aligned}$$

By Corollary D.3 and union bound we have jointly for all $j \leq p$ with probability at least

$$1 - \sum_{j=1}^{p}\exp\left(-\frac{\left(\frac{\lambda N}{4\sigma} - \sqrt{\mathrm{tr}(\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T)}\right)^2}{2\|\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T\|_2}\right)$$

that

$$\|(\epsilon^T\boldsymbol{\Phi})^{(j)}\|_2 \leq \frac{\lambda N}{4}.$$

Therefore, by Cauchy-Schwarz,

$$\begin{aligned}\epsilon^T\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star) &\leq \sum_{j=1}^{p}\|(\epsilon^T\boldsymbol{\Phi})^{(j)}\|_2\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \\ &\leq \frac{\lambda N}{4}\sum_{j=1}^{p}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2.\end{aligned}$$

This implies that

$$\frac{1}{N}\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2^2 \leq \frac{\lambda}{2}\sum_{j=1}^{p}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 + \lambda\sum_{j=1}^{p}\left(\|\boldsymbol{\beta}^{\star(j)}\|_2 - \|\hat{\boldsymbol{\beta}}^{(j)}\|_2\right).$$

Therefore

$$\begin{aligned}\frac{1}{N}\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2^2 + \frac{\lambda}{2}\sum_{j=1}^{p}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \leq & \\ \lambda\sum_{j=1}^{p}\left(\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 + \|\boldsymbol{\beta}^{\star(j)}\|_2 - \|\hat{\boldsymbol{\beta}}^{(j)}\|_2\right)\end{aligned}$$

and since $\boldsymbol{\beta}^{\star(j)} = 0$ for all $j \notin J^\star$

$$\frac{1}{N}\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2^2 + \frac{\lambda}{2}\sum_{j=1}^{p}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \leq 2\lambda \sum_{j\in J^\star} \min\left(\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2, \|\boldsymbol{\beta}^{\star(j)}\|_2\right).$$

This proves the statement. □

Lemma D.5. *Let Assumption 3.1 hold. If $\hat{\boldsymbol{\beta}}$ is a solution of Eq. (1) then we have for $\frac{\lambda N}{4\sigma} > \sqrt{\mathrm{tr}(\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T)}$ with probability at least*

$$1 - \sum_{j=1}^{p} \exp\left(-\frac{\left(\frac{\lambda N}{4\sigma} - \sqrt{\mathrm{tr}(\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T)}\right)^2}{2\|\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T\|_2}\right)$$

that

$$\sum_{j\notin J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \leq 3\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2,$$

and

$$\sum_{j=1}^{p}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \leq \frac{8\lambda m}{\bar{\kappa}^2}, \tag{D.2}$$

where

$$\bar{\kappa} := \frac{\sqrt{m}}{\sqrt{N}}\frac{\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2}{\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2}. \tag{D.3}$$

Proof. Lemma D.4 implies

$$\sum_{j=1}^{p}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \leq 4\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \tag{D.4}$$

and therefore

$$\sum_{j\notin J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \leq 3\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2,$$

which yields the first statement of Lemma D.5.
Again, by the first statement of Lemma D.4, we have with probability at least

$$1 - \sum_{j=1}^{p} \exp\left(-\frac{\left(\frac{\lambda N}{4\sigma} - \sqrt{\mathrm{tr}(\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T)}\right)^2}{2\|\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T\|_2}\right)$$

that

$$\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2 \leq \sqrt{2\lambda N}\sqrt{\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2}. \tag{D.5}$$

Therefore

$$\begin{aligned}\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 &\leq \frac{\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2}{\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2}\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2 \\ &\leq \frac{\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2}{\|\boldsymbol{\Phi}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star)\|_2}\sqrt{2\lambda N}\sqrt{\sum_{j\in J^\star}\|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2}.\end{aligned}$$

Solving this yields

$$\sum_{j\in J^\star} \|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \le \frac{2\lambda m}{\bar{\kappa}^2},$$

and by Eq. (D.4) we have

$$\begin{aligned}\sum_{j=1}^{p} \|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 &\le 4 \sum_{j\in J^\star} \|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \\ &\le \frac{8\lambda m}{\bar{\kappa}^2}.\end{aligned}$$

□

Definition D.6 (compatibility variable)**.** *Let*

$$S := \Big\{(J,b) \subset \mathcal{P}(\{1,\dots,p\}) \times (\mathbb{R}^d \backslash \{0\}) \;\Big|\; |J| \le s^\star, \sum_{j\notin J} \|b^{(j)}\|_2 \le 3 \sum_{j\in J} \|b^{(j)}\|_2 \Big\}.$$

For $n, d \in \mathbb{N}$ and $\boldsymbol{\Phi} \in \mathbb{R}^{N\times d}$ we define $\kappa(\boldsymbol{\Phi})$ by

$$\kappa(\boldsymbol{\Phi}) := \inf_{(J,b)\in S} \frac{\sqrt{m}}{\sqrt{N}} \frac{\|\boldsymbol{\Phi} b\|_2}{\sum_{j\in J} \|b^{(j)}\|_2}.$$

and call it the compatibility variable of $\boldsymbol{\Phi}$.

Remark D.7. *It holds that $\kappa \le \bar{\kappa}$.*

Corollary D.8. *Let $0 < \omega < c_1$ and let $\hat{\boldsymbol{\beta}}$ is a solution of Eq. (1) with*

$$\lambda \le \frac{\bar{\omega}\kappa^2}{8\sqrt{m}},$$

where $\bar{\omega} := \min\{\omega, c_1 - \omega\}$. Then we have for $\frac{\lambda N}{4\sigma} > \sqrt{\mathrm{tr}(\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T)}$ with probability at least

$$1 - \sum_{j=1}^{p} \exp\left(- \frac{\left(\frac{\lambda N}{4\sigma} - \sqrt{\mathrm{tr}(\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T)}\right)^2}{2\|\boldsymbol{\Phi}^{(j)}(\boldsymbol{\Phi}^{(j)})^T\|_2} \right)$$

that

$$\frac{1}{\sqrt{m}} \sum_{j=1}^{p} \|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \le \bar{\omega}. \tag{D.6}$$

If additionally Assumption 3.1 holds, then we have with the same probability for

$$\hat{J} := \Big\{ j \in \{1,\dots,p\} \;\Big|\; \|\hat{\boldsymbol{\beta}}^{(j)}\|_2 > \omega\sqrt{m} \Big\}$$

that

$$\hat{J} = J^\star.$$

Proof. The first statement follows directly from Lemma D.5.
Assume $j \in J^\star$. Then by Assumption 3.1

$$\|\hat{\boldsymbol{\beta}}^{(j)}\|_2 > \sqrt{m}\Big(c_1 - \|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2\Big) \ge \sqrt{m}(c_1 - \bar{\omega}) \ge \omega\sqrt{m},$$

which implies $J^\star \subset \hat{J}$. Assume $j \notin J^\star$, then

$$\|\hat{\boldsymbol{\beta}}^{(j)}\|_2 \le \|\hat{\boldsymbol{\beta}}^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 + \|\boldsymbol{\beta}^{\star(j)}\|_2 \le \bar{\omega}\sqrt{m} \le \omega\sqrt{m},$$

which implies $\hat{J} \subset J^\star$ □

Remark D.9. *Choosing ω optimally yields $\omega = \bar{\omega} = c_1/2$.*

Proof of Theorem 3.3. Note that $\mathbf{\Phi}^{(j)} \in \mathbb{R}^{N \times md_j}$ is block-diagonal. Since by assumption $k_j(x, x') \leq 1, \forall j \leq p$ we have

$$\operatorname{tr}\left((\mathbf{\Phi}^{(j)})^T \mathbf{\Phi}^{(j)}\right) = \operatorname{tr}\left(\mathbf{\Phi}^{(j)} (\mathbf{\Phi}^{(j)})^T\right) = \sum_{s=1}^{m} \sum_{i=1}^{n} k_j\left(x_i^{(s)}, x_i^{(s)}\right) \leq N,$$

and

$$\begin{aligned}
\|(\mathbf{\Phi}^{(j)})^T \mathbf{\Phi}^{(j)}\|_2 &= \|\mathbf{\Phi}^{(j)} (\mathbf{\Phi}^{(j)})^T\|_2 \\
&\leq \max_{s \leq m} \operatorname{tr}\left(\mathbf{\Phi}_s^{(j)} (\mathbf{\Phi}_s^{(j)})^T\right) \\
&\leq \max_{s \leq m} \sum_{i=1}^{n_s} k_j\left(x_i^{(s)}, x_i^{(s)}\right) \\
&\leq n.
\end{aligned}$$

Corollary D.8 yields the result. □

E LIFELONG ANALYSIS (PROOF OF THEOREM 4.1)

We start by proving a generic variant of Theorem 4.1, from which we can obtain the theorem in the main text as a corollary.

Theorem E.1. *Assume that the true reward functions $f_1, \ldots, f_m$ satisfy $\|f_i\|_{\mathcal{H}_{k^\star}} \leq B$ for some constant $B > 0$. Assume $\{n_s\}_{s \in \mathbb{N}}$ is a non-increasing sequence with $n_s \leq n, \forall s$. Define $N_m := \sum_{s=1}^{m} n_s$. Let ν be a distribution on $\mathcal{X}^N$ independent of $\boldsymbol{\epsilon}_1, \ldots, \boldsymbol{\epsilon}_m$. Let $V \sim \nu$ be the random vector used for forced exploration. Let $\tilde{\mathbf{\Phi}}_s \in \mathbb{R}^{N_s \times md}$ be the data matrix obtained by forced exploration. Assume the forced exploration distribution ν and $\{k_j\}_{j \leq p}$ are such that, with probability at least $1 - \delta/4$, there exists $c_\kappa > 0$ such that $\kappa(\tilde{\mathbf{\Phi}}_s) \geq c_\kappa, \forall s \leq m$. Assume further that* BASEBO *using the true kernel function for m tasks with n interactions with independent noise achieves with probability at least $1 - \delta/2$ cumulative regret lower than $R^\star(m, n)$ in the worst-case. Then, for $m_0 \in \mathbb{N}$ and $0 < \delta < 1$, if*

$$N_{m_0} \geq \frac{2n_1 \log(4mp/\delta)}{(\sqrt{N_m/m}\frac{c_1 c_\kappa^2}{32\sigma} - 1)^2},$$

with probability at least $1 - \delta$ LIBO *achieves*

$$R(m, n) \leq 2Bm_0 n + 2BN_m + R^\star(m, n).$$

Proof. Denote by $C := \{v \mid \kappa(\tilde{\mathbf{\Phi}}_s(v)) \geq c_\kappa, \forall s \leq m\}$ the set of data points such that κ is lower bounded by c_κ. By assumption we have $\mathbb{P}[V \in C] \geq 1 - \delta/4$. Denote by $\hat{J}_s \subset \{1, \ldots, p\}$ the sparsity structure predicted by LIBO after the first s tasks. Note that $\tilde{\mathbf{\Phi}}_s^{(j)} \in \mathbb{R}^{n_s m \times md_j}$ is block-diagonal. Since by assumption $k_j(x, x') \leq 1, \forall j \leq p$ we have

$$\operatorname{tr}((\tilde{\mathbf{\Phi}}_s^{(j)})^T \tilde{\mathbf{\Phi}}_s^{(j)}) = \operatorname{tr}(\tilde{\mathbf{\Phi}}_s^{(j)} (\tilde{\mathbf{\Phi}}_s^{(j)})^T) = \sum_{s=1}^{m} \sum_{i=1}^{n_s} k_j(x_i^{(s)}, x_i^{(s)}) \leq N,$$

and

$$\begin{aligned}
\|(\tilde{\mathbf{\Phi}}_s^{(j)})^T \tilde{\mathbf{\Phi}}_s^{(j)}\|_2 &= \|\tilde{\mathbf{\Phi}}_s^{(j)} (\tilde{\mathbf{\Phi}}_s^{(j)})^T\|_2 \\
&\leq \max_{s \leq m} \operatorname{tr}(\mathbf{\Phi}_s^{(j)} (\mathbf{\Phi}_s^{(j)})^T) \\
&\leq \max_{s \leq m} \sum_{i=1}^{n_s} k_j(x, x) \\
&\leq \max_{s \leq m} n_s \\
&= n_1.
\end{aligned}$$

Since V is independent of $\boldsymbol{\epsilon}_1, \dots, \boldsymbol{\epsilon}_m$, we have by Corollary D.8 for all s and $v' \in C$

$$\mathbb{P}\left[\hat{J}_s = J^\star \mid V = v'\right] \geq 1 - p \exp\left(-\frac{N_s}{2n_1}\left(\sqrt{\frac{N_s}{s}}\frac{c_1 c_\kappa^2}{32\sigma} - 1\right)^2\right).$$

By union bound and since $\frac{N_s}{s}$ is non-increasing by assumption we have for $m_0 \leq m$

$$\begin{aligned}\mathbb{P}\left[\forall m \geq s \geq m_0, \hat{J}_s = J^\star \mid V = v'\right] &\geq 1 - \sum_{s=m_0}^{m} p \exp\left(-\frac{N_s}{2n_1}\left(\sqrt{\frac{N_s}{s}}\frac{c_1 c_\kappa^2}{32\sigma} - 1\right)^2\right) \\ &\geq 1 - mp \exp\left(-\frac{N_{m_0}}{2n_1}\left(\sqrt{\frac{N_m}{m}}\frac{c_1 c_\kappa^2}{32\sigma} - 1\right)^2\right),\end{aligned}$$

where we defined $N_s \coloneqq \sum_{i=1}^{s} n_i$. If $m_0 \in \mathbb{N}$ is large enough such that

$$N_{m_0} \geq \frac{2n_1 \log(4mp/\delta)}{(\sqrt{N_m/m}\frac{c_1 c_\kappa^2}{32\sigma} - 1)^2},$$

then for all $v' \in C$

$$\mathbb{P}\left[\forall m \geq s \geq m_0, \hat{J}_s = J^\star \mid V = v'\right] \geq 1 - \delta/4.$$

By assumption we have

$$\mathbb{P}\left[V \in C\right] \geq 1 - \delta/4.$$

Because V is independent of the noise

$$\begin{aligned}\mathbb{P}\left[\exists m \geq s \geq m_0, \hat{J}_s \neq J^\star\right] &= \int \mathbb{P}\left[\exists m \geq s \geq m_0, \hat{J}_s \neq J^\star \mid V = v\right] p_V(v) dv \\ &= \int_C \mathbb{P}\left[\exists m \geq s \geq m_0, \hat{J}_s \neq J^\star \mid V = v\right] p_V(v) dv + \\ &\quad \int_{C^c} \mathbb{P}\left[\exists m \geq s \geq m_0, \hat{J}_s \neq J^\star \mid V = v\right] p_V(v) dv \\ &\leq \mathbb{P}[V \in C^c] + \mathbb{P}[V \in C]\delta/2 \\ &\leq \delta/4 + \delta/4 = \delta/2.\end{aligned}$$

For all tasks that happen after task m_0 we have jointly with probability at least $1 - \delta/2$ that $\hat{J}_s = J^\star$.
Denote by $r(k, s)$ the regret that the base bandit algorithm achieves after n interactions with kernel k in task s. By assumption $\mathbb{P}[\sum_{s=m_0}^{m} r(k^\star, s) \leq R^\star(n, m - m_0)] \geq 1 - \delta/2$. Denote by $\hat{k}_s$ the predicted kernel for task s. By union bound

$$\begin{aligned}\mathbb{P}\left[\sum_{s=m_0}^{m} r(\hat{k}_s, s) \leq \mathcal{O}(R^\star(m, n))\right] &\geq \mathbb{P}\Bigg[\forall m_0 \leq s \leq m, \hat{k}_s = k^\star \text{ and} \\ &\qquad \sum_{s=m_0}^{m} r(\hat{k}_s, s) \leq \mathcal{O}(R^\star(m - m_0, n))\Bigg] \\ &\geq 1 - \mathbb{P}\left[\sum_{s=m_0}^{m} r({k^\star}_s, s) > \mathcal{O}(R^\star(m - m_0, n))\right] \\ &\quad - \mathbb{P}\left[\exists m_0 \leq s \leq m, {k^\star}_s \neq k^\star\right] \\ &\geq 1 - \delta\end{aligned}$$

Therefore it holds with probability at least $1 - \delta$

$$R(m, n) \leq m_0 n L + L N_m + R^\star(m - m_0, n).$$

Here, the first term is an upper bound of the regret in the first m_0 tasks. The other terms are an upper bound on the reward for the other $m - m_0$ tasks. They can be divided into the regret obtained by forced exploration and the regret obtained by the base bandit task. By Lemma E.2 we know that the maximum instantaneous regret L is bounded by $2B$. Therefore

$$\begin{aligned} R(m,n) &\leq m_0 n L + L N_m + R^\star(m - m_0, n) \\ &\leq 2B m_0 n + 2B N_m + R^\star(m,n). \end{aligned}$$

□

Lemma E.2. *Let k be a kernel with $k(x,x') \leq 1, \forall x, x' \in \mathcal{X}$ and let $f \in \mathcal{H}_k$ with $\|f\|_k \leq B$, then for all $\boldsymbol{x} \in \mathcal{X}$*

$$|f(\boldsymbol{x})| \leq B.$$

Proof. By the reproducing property, we have

$$\begin{aligned} |f(\boldsymbol{x})| &= |\langle f(\cdot), k(x,\cdot)\rangle_k| \\ &\leq \|f\|_{\mathcal{H}_k} k(x,x) \\ &\leq B. \end{aligned}$$

□

A clarification is due, regarding the exact number of exploratory steps taken. In the algorithm design and in the main text, we require that during every task s, purely exploratory actions are taken at every step i where $i \leq n_s$. The number of exploratory steps has to be an integer, while the proposed rate of $n_s = \sqrt{n}/s^{1/4}$ may not be an integer. Therefore, the $i \leq n_s$ condition implies that only the first $\lfloor n_s \rfloor$ steps will be exploratory. In our proofs so far, we have assumed that at least a total of $N_{m_0} = \sum_{s=1}^{m_0} n_s$ exploratory action are chosen, which may be well larger than $\sum_{s=1}^{m_0} \lfloor n_s \rfloor$. To resolve this gap, we accumulate the non-integer remainder $n_s - \lfloor n_s \rfloor$ in a variable r. Whenever r becomes larger than 1, we increase the number forced exploration queries by 1 to $\tilde{n}_s = \lfloor n_s \rfloor + \lfloor r \rfloor$. At every task s, we force exactly $\tilde{n}_s \in \mathbb{N}$ exploratory actions, where $(\tilde{n}_1, \ldots, \tilde{n}_s)$ is calculated as described in Algorithm 5. Then to ensure that N_{m_0} exploratory datapoint are available, we calculate the smallest $\tilde{m}_0$ which satisfies:

$$\sum_{s=1}^{\tilde{m}_0} \tilde{n}_s \geq N_{m_0}$$

It is straightforward to show that by construction of Algorithm 5, $m_0 \leq \tilde{m}_0 \leq m_0 + 1$. In other words, by taking exploratory actions according to $\tilde{n}_s$ (which is an integer) we reqiure at most 1 additional task to fulfill the lower bound on the total number of required exploratory actions. In the next two corollaries we give a lower bound on the $\tilde{m}_0$ which satisfies the required dataset size N_{m_0}.

Algorithm 5 Forced Exploration Rate to Integer number of Exploratory Steps

Require: The sequence of $(n_1, \ldots, n_m)$
  $r \leftarrow 0$ ▷ r is the sum of fractional residue
  for $s \in \{1, \ldots, m\}$ **do**
    $r \leftarrow r + n_s - \lfloor n_s \rfloor$ ▷ Add the fractional part of n_s to the residue sum
    $\tilde{n}_s \leftarrow \lfloor n_s \rfloor + \lfloor r \rfloor$ ▷ If the residue sum is over 1, then add 1 to $\lfloor n_s \rfloor$
    $r \leftarrow r - \lfloor r \rfloor$
  end for
  Output: $(\tilde{n}_1, \ldots, \tilde{n}_m)$

Corollary E.3. *Assume the setting of Theorem E.1. Set the rate*

$$n_s = \sqrt{n}$$

for all $s \in \mathbb{N}$, and choose the integer number of forced exploration steps according to Algorithm 5. Then, for all $0 < \delta < 1$, with probability at least $1 - \delta$

$$R(m,n) \leq \mathcal{O}\left(B \log(mp/\delta)\sqrt{n}\right) + 2mB\sqrt{n} + R^\star(m,n).$$

Proof. Taking actions at a $n_s = \sqrt{n}$ rate via Algorithm 5, we can ensure that after $\tilde{m}_0$ many tasks the condition of Theorem E.1 on N_{m_0} is met, where

$$\frac{2\log(4mp/\delta)}{(n^{1/4}\frac{c_1 c_\kappa^2}{32\sigma}-1)^2} \leq \tilde{m}_0 \leq \frac{2\log(4mp/\delta)}{(n^{1/4}\frac{c_1 c_\kappa^2}{32\sigma}-1)^2}+1.$$

Then the proof directly from Theorem E.1 with $n_1 = \sqrt{n}$, $N_m = m\sqrt{n}$. □

Corollary E.4. *Assume the setting of Theorem E.1. Set the rate*

$$n_s = \frac{\sqrt{n}}{s^{1/4}}$$

for all $s \in \mathbb{N}$, and choose the explicit integer number of forced exploration steps according to Algorithm 5. Then, for all $0 < \delta < 1$, with probability at least $1-\delta$

$$R(m,n) \leq \mathcal{O}\left(Bn^{1/3}\log^{3/4}(mp/\delta)m^{1/3} + B\sqrt{n}m^{3/4}\right) + R^\star(m,n).$$

Proof. We have

$$N_m = \sum_{s=1}^{m}\frac{\sqrt{n}}{s^{1/4}} = \Theta\left(\sqrt{n}m^{3/4}\right)$$

Choose

$$\tilde{m}_0 = \Theta\left(\frac{\log(4mp/\delta)}{\left(\sqrt{\frac{n}{m^{1/4}}}\frac{c_1 c_\kappa}{32\sigma}-1\right)^2}\right)^{4/3}$$

and take exploratory actions according to Algorithm 5 then,

$$N_{m_0} \geq \frac{2n_1\log(4mp/\delta)}{(\sqrt{N_m/m}\frac{c_1 c_\kappa^2}{32\sigma}-1)^2}.$$

By Theorem E.1, since $n_1 = \sqrt{n}$, for all $0 < \delta < 1$, with probability at least $1-\delta$ LIBO achieves

$$\begin{aligned} R(m,n) &\leq 2Bm_0 n + 2BN_m + R^\star(m,n) \\ &\leq \mathcal{O}\left(Bn^{1/3}\log^{3/4}(mp/\delta)m^{1/3} + B\sqrt{n}m^{3/4}\right) + R^\star(m,n). \end{aligned}$$

□

E.1 BACKGROUND ON GP-UCB

To solve task s, GP-UCB first constructs *confidence sets* for $f_s(\boldsymbol{x})$ based on the history $\{(\boldsymbol{x}_{s,t}, y_{s,t})_{t\leq i}\}$ to balance exploration and exploitation at any step i. For any $\boldsymbol{x} \in \mathcal{X}$, the set $\mathcal{C}_{i-1}(\boldsymbol{x})$ defines an interval to which $f(\boldsymbol{x})$ belongs with high probability such that,

$$\mathbb{P}\left(\forall \boldsymbol{x} \in \mathcal{X} : f(\boldsymbol{x}) \in \mathcal{C}_{i-1}(\boldsymbol{x})\right) \geq 1-\delta.$$

Given a kernel k, GP-UCB builds sets of the form

$$\mathcal{C}_{i-1}(k;\boldsymbol{x}) = [\mu_{i-1}(k;\boldsymbol{x}) - \nu_i\sigma_{i-1}(k;\boldsymbol{x}),\ \mu_{i-1}(k;\boldsymbol{x}) + \nu_i\sigma_{i-1}(k;\boldsymbol{x})]$$

where the exploration coefficient ν_i depends on the desired confidence level $1-\delta$, and is often treated as a hyper-parameter of the algorithm. The functions μ_{i-1} and σ_{i-1} set the center and width of the confidence set as

$$\begin{aligned} \mu_{i-1}(k;\boldsymbol{x}) &= \boldsymbol{k}_{i-1}^T(\boldsymbol{x})(\boldsymbol{K}_{i-1} + \lambda_{\text{ucb}}^2\boldsymbol{I})^{-1}\boldsymbol{y}_{i-1} \\ \sigma_{i-1}^2(k;\boldsymbol{x}) &= k(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}_{i-1}^T(\boldsymbol{x})(\boldsymbol{K}_{i-1} + \lambda_{\text{ucb}}^2\boldsymbol{I})^{-1}\boldsymbol{k}_{i-1}(\boldsymbol{x}) \end{aligned}$$

where λ_{ucb} is a regularizer, $\boldsymbol{y}_{i-1} = [y_{s,t}]_{t<i}$ is the vector of observed values, $\boldsymbol{k}_{i-1}(\boldsymbol{x}) = [k(\boldsymbol{x}, \boldsymbol{x}_{s,t})]_{t<i}$, and $\boldsymbol{K}_{i-1} = [k(\boldsymbol{x}_{s,t}, \boldsymbol{x}_{s,t'})]_{t,t'<i}$ is the kernel matrix. GP-UCB then chooses an action that maximizes the upper confidence bound, i.e.

$$\boldsymbol{x}_{s,i} = \arg\max_{\boldsymbol{x}\in\mathcal{X}} \mu_{i-1}(\boldsymbol{x}) + \nu_i \sigma_{i-1}(\boldsymbol{x}).$$

The acquisition function balances exploring uncertain actions and exploiting the gained information via parameter ν_i. Chowdhury and Gopalan [2017] show that following this policy, and using $k^\star$ as the kernel function, yields a regret of

$$R^\star(n) = \mathcal{O}\left(Bd^\star\sqrt{n}\log\tfrac{n}{d^\star} + \sqrt{nd^\star\log\tfrac{n}{d^\star}\log\tfrac{1}{\delta}}\right).$$

E.2 LIFELONG REGRET OF GP-UCB PAIRED WITH LIBO (PROOF OF COROLLARY 4.2)

Definition E.5 (maximum information gain)**.** *The maximum information gain after t observations of GP-UCB with kernel k and parameter λ_{ucb} is defined by*

$$\gamma_t(k) = \max_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_t\in\mathcal{X}} \frac{1}{2}\log\det\left(I + \lambda_{\text{ucb}}^{-2}K_t\right),$$

where

$$(K_t)_{ij} = (\boldsymbol{\Phi}^T\boldsymbol{\Phi})_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

Theorem E.6 (Theorem 3 of Chowdhury and Gopalan [2017])**.** *Let k be a kernel and $f \in \mathcal{H}_k$, where $\mathcal{H}_k$ is the RKHS corresponding to kernel k. Let $\delta \in (0,1)$, $\|f\|_k \le B$ and assume the errors ϵ_t are conditionally σ-sub-Gaussian. Running GP-UCB with $\lambda_{\text{ucb}} = 1 + 2/n$ for n steps we have with probability at least $1-\delta$ that*

$$R(n) = \mathcal{O}(B\sqrt{n\gamma_n(k)} + \sqrt{n\gamma_n(k)(\gamma_n(k) + \log(1/\delta))})$$

Corollary E.7. *Let k_s be kernels and $f_s \in \mathcal{H}_{k_s}$, where $\mathcal{H}_{k_s}$ is the RKHS corresponding to kernel k_s. Let $\delta \in (0,1)$, $\|f_s\|_{k_s} \le B$ and assume the errors $\epsilon_{s,t}$ are i.i.d. σ-sub-Gaussian. Assume further that k_s are $\sigma(\boldsymbol{\epsilon}_1,\ldots,\boldsymbol{\epsilon}_{s-1})$-measurable. Running GP-UCB with $\lambda_{\text{ucb}} = 1 + 2/n$ for m tasks, each with n steps, we have with probability at least $1-\delta$ that jointly for all $s \in \{1,\ldots,m\}$*

$$R_s(n) = \mathcal{O}(B\sqrt{n\gamma_n(k)} + \sqrt{n\gamma_n(k)(\gamma_n(k) + \log(1/\delta))})$$

where $R_s(n)$ denotes the reward in task s after n interactions. In particular

$$R^\star(m,n) \le \mathcal{O}\left(m\sqrt{n\gamma_n(k)}(B + \sqrt{\gamma_n(k)} + \log(1/\delta))\right).$$

Proof. We will adapt the proof of Theorem 1 in Chowdhury and Gopalan [2017].
Let $\epsilon_1^s,\ldots,\epsilon_n^s$ be the noise of task s. Define a function

$$s(t) = \sum_{j=1}^{m} j\,\mathbb{1}_{\{(j-1)n+1\le t\le jn\}}$$

and a filtration on $\{1,\ldots,mn\}$

$$\mathcal{F}_t = \sigma(\epsilon_{1,1},\ldots,\epsilon_{1,n},\epsilon_{2,1},\ldots,\epsilon_{2,n},\ldots,\epsilon_{s(t),1},\ldots,\epsilon_{s(t),t-(s(t)-1)n}).$$

Further define for task s a filtration on $\{1,\ldots,n\}$

$$\mathcal{F}_t^s = \sigma(\epsilon_{s,1},\ldots,\epsilon_{s,t}).$$

Similar to the proof of Theorem 1 in Chowdhury and Gopalan [2017] define for $t \in \{1,\ldots,n\}$, $g : \mathcal{X} \to \mathbb{R}$ and $l_1,\ldots,l_n \in \mathbb{N}$

$$M_t^{g,n}(s) = \exp\left((\epsilon_{s,1:t})^T g_{1:t,l} - \frac{\sigma^2}{2}\|g_{1:t,l}\|_2^2\right)$$

where

$$g_{1:t,l} := [g(\boldsymbol{x}_1) + l_1, \dots, g(\boldsymbol{x}_t) + l_t]^T.$$

Further let $N_1, \dots, N_n$ i.i.d. with distribution $\mathcal{N}(0, \kappa)$ and independent of $\mathcal{F}_n^s$ and let h_s be a random function distributed according to the Gaussian Process measure $GP_{\mathcal{X}}(0, k_s)$ and independent of F_n^s and $N_1, \dots, N_n$. Define

$$M_t(s) = \mathbb{E}[M_t^{h_s,N}(s) \mid \mathcal{F}_n^s].$$

Now by the proof of Theorem 1 of Chowdhury and Gopalan [2017] we have that for all $s \in \{1, \dots, m\}$, $t \in \{1, \dots, n\}$ and all stooping times τ_s with respect to the filtration $\mathcal{F}_t^s$

$$\mathbb{E}[M_{\tau_s}(s)] \leq 1. \tag{E.1}$$

Given stopping times $\tau_1, \dots, \tau_m$ on $\mathcal{F}_t^1, \dots, \mathcal{F}_t^m$ we construct a stopping time τ on $\mathcal{F}_t$

$$\tau(\omega) = \min\{mn \geq t \geq 1 \mid \tau^{s(t)}(\omega) = t - (s(t) - 1)n\}. \tag{E.2}$$

We need to show that τ is a stopping time with respect to the filtration $\mathcal{F}_t$. We have

$$\{\omega \mid \tau(\omega) = t\} = \left(\bigcap_{s < s(t)} \{\omega \mid \tau_s(\omega) > n\} \right) \cap \{\omega \mid \tau^{s(t)}(\omega) = t - (s(t) - 1)n\}.$$

It holds that $\{\omega \mid \tau_s(\omega) > n\} = \{\omega \mid \tau_s(\omega) \leq n\}^c \in \mathcal{F}_n^s \subset \mathcal{F}_{sn}$ and $\{\omega \mid \tau^{s(t)}(\omega) = (s(t)-1)n - t\} \in \mathcal{F}_{t-(s(t)-1)n}^{s(t)} \subset \mathcal{F}_t$. This impies that $\{\omega \mid \tau(\omega) = t\} \in \mathcal{F}_t$ and therefore τ is a stopping time with respect to $\mathcal{F}_t$. Define

$$M_t = M_{(s(t)-1)n-t}(s(t)).$$

We have that $M_t = M_{(s(t)-1)n-t}(s(t))$ is measurable with respect to $\mathcal{F}_{(s(t)-1)n-t}^{s(t)} \subset \mathcal{F}_t$, which means M_t is $\mathcal{F}_t$-adapted. Let τ be a stopping time constructed as in Equation E.2. Then by Equation E.1

$$\mathbb{E}[M_\tau] \leq 1.$$

Define for $t \in \{1, \dots, n\}$ and $s \in \{1, \dots, m\}$

$$B_t^s(\delta) = \left\{ \omega \mid \|\epsilon_{s,1:t}\|^2_{((K_t^s + \kappa I)^{-1} + I)^{-1}} > 2 \log \left(\sqrt{\det((1+\kappa)I + K_t^s)}/\delta \right) \right\},$$

where K_t^s the design matrix for task s. Further define

$$\tau^s(\omega) = \min\{t \in \{1, \dots, n\} \mid \omega \in B_t^s(\delta)\}$$

and let τ be the corresponding stopping time on $\mathcal{F}_t$. It holds by the proof of Theorem 1 of Chowdhury and Gopalan [2017] that

$$M_t(s) = \frac{\exp\left(\frac{1}{2} \|\epsilon_{s,1:t}\|^2_{((K_t^s + \kappa I)^{-1} + I)^{-1}} \right)}{\sqrt{\det((1+\kappa)I + K_t^s)}}$$

and therefore

$$M_t = \frac{\exp\left(\frac{1}{2} \|\epsilon_{s(t),1:t-(s(t)-1)n}\|^2_{((K^{s(t)}_{t-(s(t)-1)n} + \kappa I)^{-1} + I)^{-1}} \right)}{\sqrt{\det((1+\kappa)I + K^{s(t)}_{t-(s(t)-1)n})}}.$$

Putting things together yields

$$\begin{aligned}
\mathbb{P}\left[\bigcup_{s\le m, t\le n} B_t^s(\delta)\right] &= \mathbb{P}\left[\tau \le mn\right] \\
&= \mathbb{P}\Bigg[\tau \le mn, \|\epsilon_{s(\tau),1:\tau-(s(\tau)-1)n}\|^2_{((K_\tau^{s(\tau)}+\kappa I)^{-1}+I)^{-1}} > \\
&\qquad 2\log\left(\sqrt{\det((1+\kappa)I + K^{s(\tau)}_{\tau-(s(\tau)-1)n}}/\delta\right)\Bigg] \\
&= \mathbb{P}\left[\tau \le mn, M_\tau > 1/\delta\right] \\
&\le \mathbb{P}\left[M_\tau > 1/\delta\right] \\
&\le \mathbb{E}[M_\tau]\delta = \delta.
\end{aligned}$$

Now follow the steps of the proof of Theorem 2 of Chowdhury and Gopalan [2017] and the claim follows. □

Lemma E.8. *Let $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ be a kernel with $d^{(k)} \in \mathbb{N}$ dimensional feature map and assume $k(x, x') = \boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\phi}(x') \le 1, \forall x, x' \in \mathcal{X}$. Then the maximum information gain of GP-UCB with kernel k and regularization parameter λ_{ucb} satisfies*

$$\gamma_n(k) \le \frac{1}{2}d^{(k)}\log(1 + \frac{\lambda_{\text{ucb}}^{-2}n}{d^{(k)}}).$$

Proof. This proofs follows the arguments of Vakili et al. [2021] and Kassraie et al. [2022]. We have that $K_n = \boldsymbol{\Phi}_n\boldsymbol{\Phi}_n^T$ and by the Weinstein-Aronszajn identity

$$\begin{aligned}
\frac{1}{2}\log\det(I_n + \lambda_{\text{ucb}}^{-2}K_n) &= \frac{1}{2}\log\det(I_{d^{(k)}} + \lambda_{\text{ucb}}^{-2}\boldsymbol{\Phi}_n^T\boldsymbol{\Phi}_n) \\
&\le \frac{1}{2}d^{(k)}\log\left(\operatorname{tr}(I + \lambda_{\text{ucb}}^{-2}\boldsymbol{\Phi}_n^T\boldsymbol{\Phi}_n)/d^{(k)}\right) \\
&\le \frac{1}{2}d^{(k)}\log\left(1 + \frac{\lambda_{\text{ucb}}^{-2}}{d^{(k)}}\operatorname{tr}(\boldsymbol{\Phi}_n^T\boldsymbol{\Phi}_n)\right).
\end{aligned}$$

Now

$$\begin{aligned}
\operatorname{tr}(\boldsymbol{\Phi}_n^T\boldsymbol{\Phi}_n) &= \sum_{i=1}^n \operatorname{tr}(\boldsymbol{\phi}(\boldsymbol{x}_i)\boldsymbol{\phi}(\boldsymbol{x}_i)^T) \\
&= \sum_{i=1}^n \operatorname{tr}(\boldsymbol{\phi}(\boldsymbol{x}_i)^T\boldsymbol{\phi}(\boldsymbol{x}_i)) \\
&= n
\end{aligned}$$

and therefore

$$\frac{1}{2}\log\det(I_n + \lambda_{\text{ucb}}^{-2}K_n) \le \frac{1}{2}d^{(k)}\log\left(1 + \frac{\lambda_{\text{ucb}}^{-2}n}{d^{(k)}}\right).$$

□

Corollary E.9. *Assume we are in the setting of Corollary E.3 with GP-UCB as the base bandit algorithm and $\lambda_{\text{ucb}} = 1+2/n$. Then, for all $0 < \delta < 1$, with probability at least $1-\delta$,*

$$R(m, n) = \mathcal{O}\left(Bmd^\star\sqrt{n}\log\tfrac{n}{d^\star} + m\sqrt{nd^\star\log\tfrac{n}{d^\star}\log\tfrac{1}{\delta}} + B\sqrt{n}(m + \log(mp/\delta))\right).$$

Proof. By Corollary E.7 and Lemma E.8 we have with high probability

$$R^\star(m,n) = \mathcal{O}\left(m\sqrt{n}\left((B+\log(1/\delta))\sqrt{\frac{1}{2}d^\star \log\left(1+\frac{n^3}{d^\star(n+2)^2}\right)} + \frac{1}{2}d^\star \log\left(1+\frac{n^3}{d^\star(n+2)^2}\right)\right)\right)$$
$$= \mathcal{O}\left(Bmd^\star\sqrt{n}\log\tfrac{n}{d^\star} + m\sqrt{nd^\star \log\tfrac{n}{d^\star}\log\tfrac{1}{\delta}}\right)$$

where $d^\star := d^{(k^\star)} = \sum_{j\in J^\star} d_j$. And using Corollary E.3 we have with high probability

$$R(m,n) \le \mathcal{O}\left(B\log(mp/\delta)\sqrt{n} + mB\sqrt{n}\right) + R^\star(m,n)$$
$$= \mathcal{O}\left(Bmd^\star\sqrt{n}\log\tfrac{n}{d^\star} + m\sqrt{nd^\star\log\tfrac{n}{d^\star}\log\tfrac{1}{\delta}} + Bm\sqrt{n}\right).$$

□

Corollary E.10. *Assume we are in the setting of Corollary E.4 with GP-UCB as the base bandit algorithm and* $\lambda_{\text{ucb}} = 1 + 2/n$. *Then, for all* $0 < \delta < 1$, *with probability at least* $1-\delta$,

$$R(m,n) = \mathcal{O}\left(Bmd^\star\sqrt{n}\log\tfrac{n}{d^\star} + m\sqrt{nd^\star\log\tfrac{n}{d^\star}\log\tfrac{1}{\delta}} + Bn^{1/3}\log^{3/4}(mp/\delta)m^{1/3} + B\sqrt{n}m^{3/4}\right).$$

Proof. The proof is the same as the proof for Corollary E.9, except that we use Corollary E.4 in place of Corollary E.3. □

Remark E.11. *Compare the results of Theorem 4.1 with the default alternative: not learning* k *and just setting* $\hat{k} = \sum_{j=1}^p \frac{1}{p}k_j$. *We would then only get a bound of the form*

$$R(m,n) \le \mathcal{O}\left(m\hat{B}\sqrt{n}\log(n)d\right),$$

where $B = \|f\|_{\hat{k}} = \frac{p}{s^\star}B$ *and* $d = \sum_{j=1}^p d_j \ge n$, *which is not sublinear in* n.

E.3 FORCED EXPLORATION LOWER BOUND (PROOF OF PROPOSITION 4.3)

Assumption E.12. *Assume there exists* $c_c, c_{od} > 0$

$$\frac{m}{N}(\mathbf{\Phi}^T\mathbf{\Phi})_{i,i} \ge c_d, \qquad \forall i \in \{1,\dots,md\}$$

and

$$\frac{m}{N}(\mathbf{\Phi}^T\mathbf{\Phi})_{i,j} < c_{od}, \qquad \forall i \ne j \in \{1,\dots,md\}.$$

Lemma E.13. *Let Assumption E.12 be satisfied. Then* $\kappa \ge \sqrt{c_d/s^\star - 5c_{od}}$.

Proof. Let $(b,J) \in S$. We have by definition of S

$$\left(\frac{\sqrt{m}}{\sqrt{N}}\frac{\|\mathbf{\Phi}\boldsymbol{b}\|_2}{\sum_{j\in J}\|b^{(j)}\|_2}\right)^2 = \frac{m}{N}\frac{b^T(\mathbf{\Phi}^T\mathbf{\Phi})b}{\left(\sum_{j\in J}\|b^{(j)}\|_2\right)^2}$$
$$\ge \frac{m}{N}\frac{\sum_{s=1}^m\sum_{i,j\in J}\boldsymbol{b}_s^{(i)}(\mathbf{\Phi}^T\mathbf{\Phi})_{i,j}\boldsymbol{b}_s^{(j)}}{\left(\sum_{j\in J}\sqrt{\sum_{s=1}^m(\boldsymbol{b}_s^{(j)})^2}\right)^2}$$
$$- 4\frac{m}{N}\frac{\sum_{s=1}^m\sum_{i\in J,j\in J^c}|\boldsymbol{b}_s^{(i)}(\mathbf{\Phi}^T\mathbf{\Phi})_{i,j}\boldsymbol{b}_s^{(j)}|}{\left(\sum_{j\in J}\sqrt{\sum_{s=1}^m(\boldsymbol{b}_s^{(j)})^2}\right)\left(\sum_{j\notin J}\sqrt{\sum_{s=1}^m(\boldsymbol{b}_s^{(j)})^2}\right)}$$

By Assumption E.12

$$\frac{m}{N}\frac{\sum_{s=1}^m \sum_{i,j\in J} \boldsymbol{b}_s^{(i)} (\boldsymbol{\Phi}^T\boldsymbol{\Phi})_{i,j} \boldsymbol{b}_s^{(j)}}{\left(\sum_{j\in J}\sqrt{\sum_{s=1}^m (\boldsymbol{b}_s^{(j)})^2}\right)^2} \geq \sum_{s=1}^m \sum_{i\in J} \frac{\boldsymbol{b}_s^{(i)} c_2 \boldsymbol{b}_s^{(i)}}{\left(\sum_{j\in J}\sqrt{\sum_{s=1}^m (\boldsymbol{b}_s^{(j)})^2}\right)^2} - \frac{\sum_{s=1}^m \sum_{i\neq j, i,j\in J} |\boldsymbol{b}_s^{(i)} c_{od} \boldsymbol{b}_s^{(j)}|}{\left(\sum_{j\in J}\sqrt{\sum_{s=1}^m (\boldsymbol{b}_s^{(j)})^2}\right)^2}.$$

Since for $q > 0$ using $\|\cdot\|_1 \leq \sqrt{s}\|\cdot\|_2$

$$\sum_{s=1}^m \sum_{i\in J} \frac{\boldsymbol{b}_s^{(i)} c_2 \boldsymbol{b}_s^{(i)}}{\left(\sum_{j\in J}\sqrt{\sum_{s=1}^m (\boldsymbol{b}_s^{(j)})^2}\right)^2} \geq c_2 \sum_{s=1}^m \sum_{i\in J} \frac{(\boldsymbol{b}_s^{(i)})^2}{s^\star \left(\sqrt{\sum_{j\in J}\sum_{s=1}^m (\boldsymbol{b}_s^{(j)})^2}\right)^2} = \frac{c_2}{s^\star}$$

and using Cauchy-Schwarz to prove

$$\sum_{k,l} (x_k y_l)^2 \geq \sum_{k,l} (x_k y_l)(y_k x_l)$$

which implies

$$\sqrt{\sum_{k,l} (x_k y_l)^2} \geq \sum_l |x_l y_l|$$

we get

$$\begin{aligned}
\frac{\sum_{s=1}^m \sum_{i\neq j, i,j\in J} |\boldsymbol{b}_s^{(i)} c_{od} \boldsymbol{b}_s^{(j)}|}{\left(\sum_{j\in J}\sqrt{\sum_{s=1}^m (\boldsymbol{b}_s^{(j)})^2}\right)^2} &= c_{od} \frac{\sum_{s=1}^m \sum_{i\neq j, i,j\in J} |\boldsymbol{b}_s^{(i)}||\boldsymbol{b}_s^{(j)}|}{\sum_{i,j\in J}\sqrt{\sum_{k,l} (\boldsymbol{b}_l^{(i)} \boldsymbol{b}_k^{(j)})^2}} \\
&\leq c_{od} \frac{\sum_{s=1}^m \sum_{i\neq j, i,j\in J} |\boldsymbol{b}_s^{(i)}||\boldsymbol{b}_s^{(j)}|}{\sum_{i,j\in J}\sum_{s=1}^m |\boldsymbol{b}_s^{(i)}||\boldsymbol{b}_s^{(j)}|} \\
&\leq c_{od}.
\end{aligned}$$

Also

$$\begin{aligned}
&\frac{\sum_{s=1}^m \sum_{i\in J, j\in J^c} |\boldsymbol{b}_s^{(i)} c_{od} \boldsymbol{b}_s^{(j)}|}{\left(\sum_{j\in J}\sqrt{\sum_{s=1}^m (\boldsymbol{b}_s^{(j)})^2}\right)\left(\sum_{j\notin J}\sqrt{\sum_{s=1}^m (\boldsymbol{b}_s^{(j)})^2}\right)} \\
&= c_{od} \frac{\sum_{s=1}^m \sum_{i\in J, j\in J^c} |\boldsymbol{b}_s^{(i)}||\boldsymbol{b}_s^{(j)}|}{\sum_{i\in J, j\in J^c}\sqrt{\sum_{k,l} (\boldsymbol{b}_l^{(i)} \boldsymbol{b}_k^{(j)})^2}} \\
&= c_{od} \frac{\sum_{s=1}^m \sum_{i\in J, j\in J^c} |\boldsymbol{b}_s^{(i)}||\boldsymbol{b}_s^{(j)}|}{\sum_{i\in J, j\in J^c}\sum_{s=1}^m |\boldsymbol{b}_s^{(i)} \boldsymbol{b}_s^{(j)}|} \\
&= c_{od}.
\end{aligned}$$

Therefore

$$\frac{\sqrt{m}}{\sqrt{N}} \frac{\|\boldsymbol{\Phi}\boldsymbol{b}\|_2}{\sum_{j\in J}\|b^{(j)}\|_2} \geq \sqrt{c_2/s - 5c_{od}}.$$

□

Proposition E.14. *Let μ be the Lebesgue measure and $d = p$. Assume that $\boldsymbol{\phi}_i \in L^2_\mu(\mathcal{X})$, $i \in \{1, \dots, p\}$ are orthogonal and satisfy $\|\boldsymbol{\phi}_i\|_{L^2_\mu(\mathcal{X})}/Vol(\mathcal{X}) \geq z$, for all $i \in \{1, \dots, p\}$. Assume also that $k_i(x, x) = \boldsymbol{\phi}_i(\boldsymbol{x})^2 \leq 1$ for all $\boldsymbol{x} \in \mathcal{X}$. Choose $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ i.i.d. uniformly from $\mathcal{X}$ and let*

$$\boldsymbol{\Phi}_s \coloneqq \begin{bmatrix} \boldsymbol{\phi}(\boldsymbol{x}_1) & \dots & \boldsymbol{\phi}(\boldsymbol{x}_n) \end{bmatrix}^T \in \mathbb{R}^{n \times d} \qquad \forall s \leq m.$$

Then with probability at least $1 - \delta$ Assumption E.12 is satisfied with

$$c_d = \left(z - \sqrt{\frac{1}{2n} \log(4d/\delta)} \right)$$

and

$$c_{od} = \sqrt{\frac{2}{n} \log\left(\frac{4d^2}{\delta}\right)}.$$

Proof. For the second, let X be a random variable uniformly distributed on $\mathcal{X}$ and denote by $v_i \coloneqq \boldsymbol{\phi}_i(\boldsymbol{x}_{1:n})$ the ith column of $\boldsymbol{\Phi}_s$. It holds that

$$\mathbb{E}[\boldsymbol{\phi}_i(\boldsymbol{x})^2] = \frac{1}{\mathrm{Vol}(\mathcal{X})} \int_{\mathcal{X}} \boldsymbol{\phi}_i(\boldsymbol{x})^2 \mathrm{d}\mu(\boldsymbol{x}) \geq z, \quad \forall i \leq d.$$

Therefore

$$\mathbb{E}[\|v_i\|_2^2] = \mathbb{E}\left[\sum_{i=1}^n \boldsymbol{\phi}_i(\boldsymbol{x}_i)^2\right] \geq nz.$$

By union bound and Höffding's inequality

$$\mathbb{P}\left[\exists i \leq d, \left|\|v_i\|_2^2 - \mathbb{E}\left[\|v_i\|_2^2\right]\right| \geq \epsilon\right] \leq 2d \exp(-\frac{2\epsilon^2}{n})$$

or

$$\mathbb{P}\left[\exists i \leq d, \left|\|v_i\|_2^2 - \mathbb{E}\left[\|v_i\|_2^2\right]\right| \geq \sqrt{\frac{n}{2} \log(\frac{4d}{\delta})}\right] \leq \delta.$$

Therefore with probability at least $1 - \delta/2$ for all $i \leq d$

$$\|v_i\|_2^2 \geq \mathbb{E}\left[\|v_i\|_2^2\right] - \sqrt{\frac{n}{2} \log(\frac{4d}{\delta})} \geq nz - \sqrt{\frac{n}{2} \log(\frac{4d}{\delta})}.$$

Further, for $i \neq j$

$$\mathbb{E}[\boldsymbol{\phi}_i(\boldsymbol{x})\boldsymbol{\phi}_j(\boldsymbol{x})] = \frac{1}{\mathrm{Vol}(\mathcal{X})} \int_{\mathcal{X}} \boldsymbol{\phi}_i(\boldsymbol{x})\boldsymbol{\phi}_j(\boldsymbol{x}) \mathrm{d}\mu(\boldsymbol{x}) = 0,$$

since $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$ are orthogonal in $L^2_\mu(\mathcal{X})$. By assumption $\boldsymbol{\phi}_i(\boldsymbol{x}) \leq 1, \forall i \leq d, \forall \boldsymbol{x} \in \mathcal{X}$ and by Höffding's inequality

$$\mathbb{P}[|\langle v_i, v_j \rangle| \geq \epsilon] \leq 2 \exp(-\frac{\epsilon^2}{2n}).$$

and therefore for $0 \leq \delta \leq 1$

$$\mathbb{P}\left[\exists i \neq j, |\langle v_i, v_j \rangle| \geq \sqrt{2n \log\left(\frac{4d^2}{\delta}\right)}\right] \leq \delta/2.$$

We derived that with probability at least $1 - \delta$

$$(\boldsymbol{\Phi}^T \boldsymbol{\Phi})_{ii}/n \geq c_2 = \left(z - \sqrt{\log(4d/\delta)/2n}\right)$$

and for $i \neq j$

$$(\boldsymbol{\Phi}^T \boldsymbol{\Phi})_{ij}/n < \sqrt{\frac{2}{n} \log\left(\frac{4d^2}{\delta}\right)}.$$

□

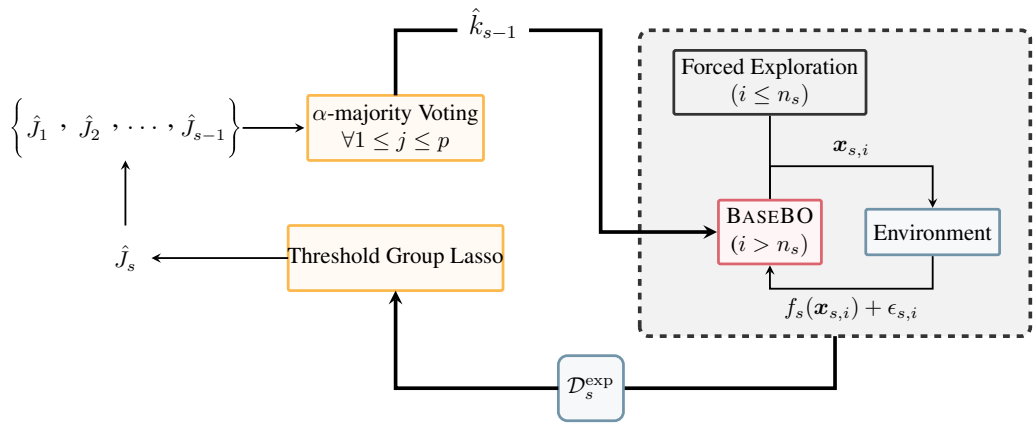

Figure 4: F-LIBO visualized. The yellow boxes corresponds to modules of F-META-KGL.

Corollary E.15. *Assume the setting of Proposition E.14. Then*

$$\kappa \geq \sqrt{z/s^\star - \sqrt{\frac{\log(4d/\delta)/(2s^\star) + 50\log(4d^2/\delta)}{n}}} = \mathcal{O}(1).$$

F FEDERATED ANALYSIS (PROOF OF THEOREM 5.1)

Recall that in the federated setting, each client minimizes the following loss locally.

$$\begin{aligned}\hat{\boldsymbol{\beta}}_{s,\text{prvt}} &:= \arg\min_{\boldsymbol{\beta}_s \in \mathbb{R}^d} \mathcal{L}\left(\boldsymbol{\beta}_s; \mathcal{D}_s^{\text{exp}}\right) \\ &= \arg\min_{\boldsymbol{\beta}_s \in \mathbb{R}^d} \frac{1}{n_s}\|\boldsymbol{y}_s - \boldsymbol{\Phi}_s\boldsymbol{\beta}_s\|_2^2 + \lambda \sum_{j=1}^{p} \|{\boldsymbol{\beta}_s}^{(j)}\|_2.\end{aligned} \tag{F.1}$$

In this section, for simplicity we refer to the solution as $\hat{\boldsymbol{\beta}}_s$. We may further omit the subscript s, whenever it can be determined from the context. For our federated analysis, we require a slightly stronger version of the Beta-min assumption.

Assumption F.1 (Beta-min federated)**.** *Assume there exists $c_{1,\text{f}} > 0$ such that for all $s \leq m$ and $j \in J^\star$*

$$\|\boldsymbol{\beta}_s^{\star(j)}\|_2 \geq c_{1,\text{f}}.$$

Remark F.2. *Note that Assumption F.1 implies Assumption 3.1.*

F.1 CONSISTENCY OF THE META-LEARNED KERNEL

In this section we prove the equivalent of Theorem 3.3 in the federated setting.

Theorem F.3 (Consistency of F-META-KGL)**.** *Let $\omega \in (0, c_{1,\text{f}})$ and $\bar{\omega} := \min\{\omega, c_{1,\text{f}} - \omega\}$. Let Assumption F.1 hold. Let $n_s = \underline{n}, \forall s \leq m$ and assume $\underline{n}$ is large enough to satisfy $\bar{\omega} > (\sqrt{\log(p/\bar{\alpha})} + 1)32\sigma/(\sqrt{\underline{n}}c_\kappa^2)$, where $\bar{\alpha} := \max\{\alpha, 1-\alpha\}$. Assume that $\boldsymbol{\Phi}_s \in \mathbb{R}^{n\times d}$ satisfy Assumption 3.2 with c_κ for $s = 1, \ldots, m$. Let $\hat{\boldsymbol{\beta}}$ be a solution of Equation F.1 with regularization parameter $\lambda = \bar{\omega}c_\kappa^2/8$. Then $\hat{J}_\text{f}$ is a consistent estimator in $\underline{n}$ and m, that is*

$$\lim_{\underline{n}\to\infty} \mathbb{P}\left[\hat{J}_\text{f} = J^\star\right] = 1 \qquad \textit{and} \qquad \lim_{m\to\infty} \mathbb{P}\left[\hat{J}_\text{f} = J^\star\right] = 1.$$

We start by proving the necessary lemmas.

Lemma F.4. *Let $\hat{\boldsymbol{\beta}}$ is a solution of equation F.1 and*

$$\lambda \leq \frac{\bar{\omega}\kappa^2}{8},$$

where $\bar{\omega} := \min\{\omega, c_{1,\mathrm{f}} - \omega\}$ *for* $0 < \omega < c_{1,\mathrm{f}}$. *Then we have for* $\frac{\lambda n_s}{4\sigma} > \sqrt{\mathrm{tr}(\Phi_s^{(j)}(\Phi_s^{(j)})^T)}$ *with probability at least*

$$1 - p \max_{j \le p} \exp\left(- \frac{\left(\frac{\lambda n_s}{4\sigma} - \sqrt{\mathrm{tr}(\Phi_s^{(j)}(\Phi_s^{(j)})^T)}\right)^2}{2\|\Phi_s^{(j)}(\Phi_s^{(j)})^T\|_2} \right)$$

that

$$\sum_{j=1}^{p} \|\hat{\boldsymbol{\beta}}_s^{(j)} - \boldsymbol{\beta}_s^{\star(j)}\|_2 \le \bar{\omega}. \tag{F.2}$$

If additionally Assumption F.1 holds, then we have with the same probability for

$$\hat{J}_{s,\mathrm{f}} = \left\{ j \in \{1, \dots, p\} \;\middle|\; \|\hat{\boldsymbol{\beta}}_s^{(j)}\|_2 > \omega \right\}$$

that

$$\hat{J}_{s,\mathrm{f}} = J^\star.$$

Proof. Follows directly from Corollary D.8 with $m = 1$. □

Lemma F.5 (Chernoff-Höffding bound)**.** *Let* $X_1, \dots, X_n$ *be i.i.d Bernoulli random variables with* $\mathbb{E}[X_i] = p_i$. *Define* $p := \frac{1}{n}\sum_{i=1}^n p_i$, *then for* $t < np$,

$$\begin{aligned}\mathbb{P}\left[\sum_{i=1}^n X_i \le t\right] &\le \exp\left(-n\left(\frac{t}{n}\log\left(\frac{t}{np}\right) + (1 - t/n)\log\left(\frac{1 - t/n}{1 - p}\right)\right)\right) \\ &\le \exp\left(-n\frac{(p - t/n)^2}{2p(1-p)}\right).\end{aligned}$$

Lemma F.6. *Let* $0 < w < c_{1,\mathrm{f}}$ *and let* $\hat{\boldsymbol{\beta}}_s$ *be the solution of equation F.1 for tasks* $s \le m$ *and* $\lambda \le \frac{\bar{\omega}\kappa^2}{8}$. *Define for* $j \in \{1, \dots, p\}$

$$Q_j = \left\{ s \in \{1, \dots, m\} \;\middle|\; \|\hat{\boldsymbol{\beta}}_s^{(j)}\|_2 > w \right\}$$

and for $\alpha > 0$

$$\hat{J}_{\mathrm{f}} = \left\{ j \in \{1, \dots, p\} \;\middle|\; |Q_j| > m\alpha \right\}. \tag{F.3}$$

Define for $s \in \{1, \dots, m\}$

$$v_s := 1 - p \max_{j \le p} \exp\left(- \frac{\left(\frac{\lambda n_s}{4\sigma} - \sqrt{\mathrm{tr}(\Phi_s^{(j)}(\Phi_s^{(j)})^T)}\right)^2}{2\|\Phi_s^{(j)}(\Phi_s^{(j)})^T\|_2} \right)$$

and

$$v := \frac{1}{m}\sum_{s=1}^m v_s.$$

Assume that $\frac{\lambda n_s}{4\sigma} > \sqrt{\mathrm{tr}(\Phi_s^{(j)}(\Phi_s^{(j)})^T)}, \forall s \le m$ *and* $v > \bar{\alpha} := \min\{\alpha, 1 - \alpha\}$. *Then*

$$\mathbb{P}\left[J^\star = \hat{J}_{\mathrm{f}}\right] \ge 1 - p\exp\left(-m\frac{(v - \bar{\alpha})^2}{2v(1-v)}\right).$$

Proof. Recall that $\boldsymbol{\epsilon}_s = [\epsilon_{s,i}]_{i=1}^n$. Since $\boldsymbol{\epsilon}_1, \dots, \boldsymbol{\epsilon}_m$ are independent,

$$\mathbb{1}_{\left\{\sum_{j=1}^p \|\hat{\boldsymbol{\beta}}_1^{(j)} - \boldsymbol{\beta}_1^{\star(j)}\|_2 \le \bar{\omega}\right\}}, \dots, \mathbb{1}_{\left\{\sum_{j=1}^p \|\hat{\boldsymbol{\beta}}_m^{(j)} - \boldsymbol{\beta}_m^{\star(j)}\|_2 \le \bar{\omega}\right\}}$$

are independent and Bernoulli distributed with coefficient

$$\mathbb{P}\left[\sum_{s=1}^p \|\hat{\boldsymbol{\beta}}_s^{(j)} - \boldsymbol{\beta}_s^{\star(j)}\|_2 \le \bar{\omega}\right] \ge v_s,$$

where we used Lemma F.4 and set $\bar{\omega} := \min\{\omega, c_{1,\mathrm{f}} - \omega\}$. If $j \in J^\star$ and $\|\hat{\boldsymbol{\beta}}_s^{(j)} - \boldsymbol{\beta}_s^{\star(j)}\|_2 < \bar{\omega}$, then by Assumption F.1

$$\|\hat{\boldsymbol{\beta}}_s^{(j)}\|_2 > c_{1,\mathrm{f}} - \|\hat{\boldsymbol{\beta}}_s^{(j)} - \boldsymbol{\beta}^{\star(j)}\|_2 \ge c_{1,\mathrm{f}} - \bar{\omega} \ge \omega,$$

which implies $J^\star \subset \hat{J}$. If $j \notin J^\star$ and $\|\hat{\boldsymbol{\beta}}_s^{(j)} - \boldsymbol{\beta}_s^{\star(j)}\|_2 < \bar{\omega}$, then

$$\|\hat{\boldsymbol{\beta}}_s^{(j)}\|_2 \le \|\hat{\boldsymbol{\beta}}_s^{(j)} - \boldsymbol{\beta}_s^{\star(j)}\|_2 + \|\boldsymbol{\beta}_s^{\star(j)}\|_2 \le \bar{\omega} \le \omega.$$

We have by Lemma F.5 and for $v > \bar{\alpha}$,

$$\begin{aligned}
\mathbb{P}\left[\hat{J}_{\mathrm{f}} = J^\star\right] &\ge \mathbb{P}\left[\forall j \in J^\star, |Q_j| \ge m/x; \forall j \notin J^\star, |Q_j| < m/x\right] \\
&\ge \mathbb{P}\left[\forall j \notin J^\star, \sum_{s=1}^m \mathbb{1}_{\left\{\|\hat{\boldsymbol{\beta}}_s^{(j)}\|_2 \le w\right\}} \ge m/x; \forall j \in J^\star, \sum_{s=1}^m \mathbb{1}_{\left\{\|\hat{\boldsymbol{\beta}}_s^{(j)}\|_2 > w\right\}} \ge m/x\right] \\
&\ge \mathbb{P}\left[\forall j \in \{1, \dots, p\}, \sum_{s=1}^m \mathbb{1}_{\left\{\|\hat{\boldsymbol{\beta}}_s^{(j)} - \boldsymbol{\beta}_s^{\star(j)}\|_2 \le \bar{\omega}\right\}} \ge m \min\{\alpha, 1-\alpha\}\right] \\
&\ge \mathbb{P}\left[\sum_{s=1}^m \mathbb{1}_{\left\{\sum_{j=1}^p \|\hat{\boldsymbol{\beta}}_s^{(j)} - \boldsymbol{\beta}_s^{\star(j)}\|_2 \le \bar{\omega}\right\}} \ge m\bar{\alpha}\right] \\
&\ge 1 - \exp\left(-m\left(\bar{\alpha}\log\left(\frac{\bar{\alpha}}{v}\right) + (1-\bar{\alpha})\log\left(\frac{1-\bar{\alpha}}{1-v}\right)\right)\right) \\
&\ge 1 - \exp\left(-m\frac{(v-\bar{\alpha})^2}{2v(1-v)}\right).
\end{aligned} \tag{F.4}$$

□

Proof of Theorem F.3. Assume the setting of Lemma F.6 and that there exists $c_\kappa > 0$ such that $\kappa \ge c_\kappa$. Set $\lambda = \frac{\bar{\omega}c_\kappa^2}{8}$, $n_s = \underline{n}, \forall s \le m$ and assume $\frac{\lambda\sqrt{\underline{n}}}{4\sigma} > 1$ and $v = 1 - p\exp(-(\lambda\sqrt{\underline{n}}/4\sigma - 1)^2/2) > \bar{\alpha}$.

Note that $\Phi_s^{(j)} \in \mathbb{R}^{N \times md_j}$ is block-diagonal. Since by assumption $k_j(x, x') \le 1, \forall j \le p$, we have

$$\|(\Phi_s^{(j)})^T\Phi_s^{(j)}\|_2 \le \mathrm{tr}((\Phi_s^{(j)})^T\Phi_s^{(j)}) = \mathrm{tr}(\Phi_s^{(j)}(\Phi_s^{(j)})^T) = \sum_{i=1}^{\underline{n}} k_j\left(x_i^{(s)}, x_i^{(s)}\right) \le \underline{n}.$$

Lemma F.6 yields the result. □

F.2 LIFELONG REGRET OF F-LIBO (PROOF OF THEOREM 5.1)

We start by stating Theorem 5.1 more rigorously.

Theorem F.7. *Assume that the true reward functions $f_1, \dots, f_m$ satisfy $\|f_i\|_{\mathcal{H}_{k^\star}} \le B$ for some constant $B > 0$. Let $\bar{n}$ be the number of times forced exploration is used in each task. Let ν be a distribution on $\mathcal{X}^{\bar{n}m}$ independent of $\boldsymbol{\epsilon}_1, \dots, \boldsymbol{\epsilon}_m$. Let $V \sim \nu$ be the random vector used for forced exploration. Let $\tilde{\Phi}_s \in \mathbb{R}^{\bar{n} \times md}$ be the data matrix obtained by forced exploration in task s. Set $\lambda = \bar{\omega}c_\kappa^2/8$. Assume the forced exploration distribution ν and $\{k_j\}_{j \le p}$ are such that, with probability at least $1 - \delta/4$, there exists $c_\kappa > 0$ such that $\kappa(\tilde{\Phi}_s) \ge c_\kappa, \forall s \le m$. Assume further that the base bandit algorithm using the true*

kernel function achieves on m tasks with independent noise with probability at least $1-\delta/2$ cumulative regret lower than $R^\star(n,m)$. Define

$$v := 1 - p\exp\left(-\frac{1}{2}\left(\frac{\bar{\omega}c_\kappa^2\sqrt{\bar{n}}}{32\sigma} - 1\right)^2\right).$$

and assume for all $s \le m$

$$v \ge \bar{\alpha}, \qquad \frac{\bar{\omega}c_\kappa^2\sqrt{\bar{n}}}{32\sigma} > 1. \tag{F.5}$$

Then with probability at least $1-\delta$, LIBO *(using* F-META-KGL *to predict the kernel) achieves*

$$R(m,n) \le \mathcal{O}\left(Bn\log(mp/\delta)/\bar{n} + Bm\bar{n}\right) + R^\star(n,m).$$

Proof. Similar to the proof of Theorem E.1 we have by Equation F.4 for all s and $v' \in C$

$$\mathbb{P}\left[\hat{J}_s = J^\star \mid V = v'\right] \ge 1 - p\exp\left(-s\left(\bar{\alpha}\log\left(\frac{\bar{\alpha}}{v}\right) + (1-\bar{\alpha})\log\left(\frac{1-\bar{\alpha}}{1-v}\right)\right)\right).$$

By union bound we have for $m_0 \le m$

$$\begin{aligned}
&\mathbb{P}\Big[\forall m \ge s \ge m_0, \hat{J}_s = J^\star \mid V = v'\Big] \\
&\qquad \ge 1 - \sum_{s=m_0}^{m} p\exp\left(-s\left(\bar{\alpha}\log\left(\frac{\bar{\alpha}}{v}\right) + (1-\bar{\alpha})\log\left(\frac{1-\bar{\alpha}}{1-v}\right)\right)\right) \\
&\qquad \ge 1 - mp\exp\left(-m_0\left(q - (1-\bar{\alpha})\log\left(1-v\right)\right)\right),
\end{aligned}$$

where $q := \bar{\alpha}\log\left(\bar{\alpha}\right) + (1-\bar{\alpha})\log\left(1-\bar{\alpha}\right)$. Set

$$m_0 = \left\lceil \frac{\log(4mp/\delta)}{\bar{q} + (1-\bar{\alpha})(\bar{w}c_\kappa^2\sqrt{\bar{n}}/32\sigma - 1)^2/2} \right\rceil,$$

where $\bar{q} := q - (1-\bar{\alpha})\log(p)$. Following the same steps as in the proof of Theorem E.1 we get

$$\begin{aligned}
R(m,n) &\le \mathcal{O}\left(m_0 nL + Lm\bar{n}\right) + R^\star(n, m-m_0) \\
&\le \mathcal{O}\left(2Bm_0 n + 2Bm\bar{n}\right) + R^\star(n,m) \\
&\le \mathcal{O}\left(Bn\log(mp/\delta)/\bar{n} + Bm\bar{n}\right) + R^\star(n,m).
\end{aligned}$$

□

Corollary F.8. *Assume the setting of Theorem F.7 and set $\bar{n} = \sqrt{n}$. Then with probability at least $1-\delta$ we have*

$$R(m,n) \le \mathcal{O}\left(B\sqrt{n}(\log(mp/\delta) + m)\right) + R^\star(n,m).$$

F.3 PERFORMANCE OF GP-UCB PAIRED WITH F-LIBO

Corollary F.9. *Assume we are in the setting of Corollary F.8 with GP-UCB as the base bandit algorithm and $\lambda_{\text{ucb}} = 1 + 2/n$. Then, for all $0 < \delta < 1$, with probability at least $1-\delta$,*

$$R(m,n) = \mathcal{O}\left(Bmd^\star\sqrt{n}\log\tfrac{n}{d^\star} + m\sqrt{nd^\star\log\tfrac{n}{d^\star}\log\tfrac{1}{\delta}} + B\sqrt{n}(m + \log(mp/\delta))\right).$$

Proof. The proof is the same as the proof for Corollary E.9, except that we use Corollary F.8 in place of Corollary E.3. □

G EXPERIMENT DETAILS

For the synthetic experiments, we initiate the algorithms with $\omega = c_1/2$. For all experiments, the exploration coefficient of the GP-UCB algorithm is set to $\nu_i = 10$ and $\lambda_{\text{ucb}} = 0.1$. Experiment are all repeated 20 times for difference random seeds, and the plots show the corresponding standard error. The remaining experiment settings are detailed in the following subsections.

G.1 OFFLINE DATA EXPERIMENTS

We generate the reward functions $f_1, ..., f_{30}$ from the synthetic environment. Corresponding to each f_s, we generate a data set $\mathcal{D}_s$ of size $n = 10$ by sampling points $\boldsymbol{x}_{s,1}, \ldots, \boldsymbol{x}_{s,n}$ i.i.d from a uniform distribution $\mathcal{U}(\mathcal{X})$ over the domain $\mathcal{X} = [0, 1]$ and collecting the corresponding noisy function values $y_{s,i} = f_s(\boldsymbol{x}_{s,i}) + \epsilon$, where the noise is samples from $\mathcal{N}(0, \sigma^2 = 0.01)$. We initiate META-KGL with the lasso regularization parameter of $\lambda = 0.25$ and F-META-KGL with $\lambda = 0.015$. For F-META-KGL, we set the majority vote threshold to $\alpha = 0.25$.

G.2 LIFELONG DATA EXPERIMENTS

For experiments using synthetic data, we set $n = 100$, and for the experiments on GLMNET data, there are $n = 144$ BO steps in each task. To run LIBO on the synthetic environment we set $\lambda = 0.5$ and for F-LIBO we set $\lambda = 0.2$. On the GMLNET environment, we instantiate LIBO with $\omega = 0.25$ and $\lambda = 0.015$, and F-META-KGL with $\alpha = 0.25$, $\omega = 10^{-6}$, $\lambda = 2.6 \times 10^{-6}$.

G.3 FURTHER EXPERIMENTS WITH SYNTHETIC DATA

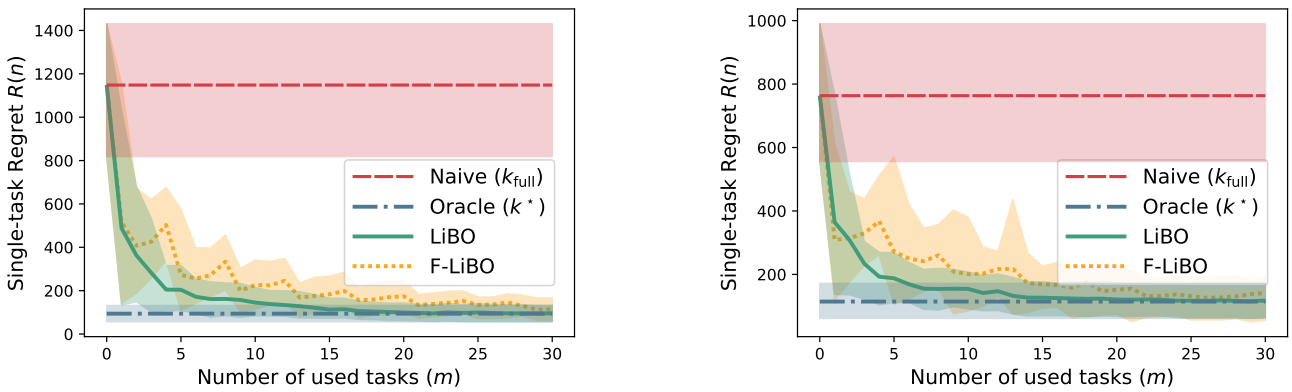

Figure 5: Single task cumulative regret of GP-UCB with meta-learned kernel $\hat{k}$ on an increasing number of meta-training tasks. Left: base kernels constructed with 2-dimensional cosine basis. Right: base kernels constructed with 1-dimensional Legendre polynomials. The BO performance with meta-leaned kernels quickly approaches oracle performance as the number of meta-training task increases.

Offline Data Analogous to the offline data experiments in Section 6.1, we provide additional results for a two-dimensional domain and Legendre polynomials instead of cosine bases in Figure 5. In particular, the left plot corresponds to $\mathcal{X} = [0, 1]^2$ as the domain and the first 50 2-dimensional cosine basis functions, i.e., $\phi_{i,j}(x) = \cos(i\pi x_1)\cos(\pi x_2), \forall x \in \mathcal{X}$, as the feature maps. For the right plot we choose $\mathcal{X} = [-1, 1]$ as the domain and use the first 50 Legendre Polynomials as the feature maps.

Figure 5 shows that both meta-learners converge with increasing number of tasks to the oracle kernel. This holds for different sets of base kernels and kernels with more than 1 input dimension. This empirically validates the theoretical findings of Theorem 3.3 and Theorem F.3. Somewhat peculiar is that we can observe oscillating behavior for the federated algorithm (yellow). This is a result of discrete nature of the voting system. The the total of number of tasks is a multiple of α the value $|\hat{J}_s|$ is large, while for points directly after that $|\hat{J}_s|$ are small. With increasing number of tasks the discretization has a lesser impact on the kernel estimation and the amplitude of the oscillations decreases.

Lifelong Data We now present modifications of the lifelong BO experiments in Section 6.2. In particular, we consider other base kernels as well as a modification of LIBO where we use all collected data for meta-learning $\hat{k}$ instead of only the forced exploration data $\mathcal{D}_{1:s}^{\text{exp}}$. The results are depicted in Figure 6. Figure **(a)** and **(b)** correspond to 50 cosine basis functions as feature maps for the base kernels. For Figure **(c)** and **(d)**, we use the first 50 Legendre polynomials as feature maps. The plots on the left (i.e. Fig. a, c) are generated with LIBO and F-LIBO, as presented in Algorithm 2 and 3, where

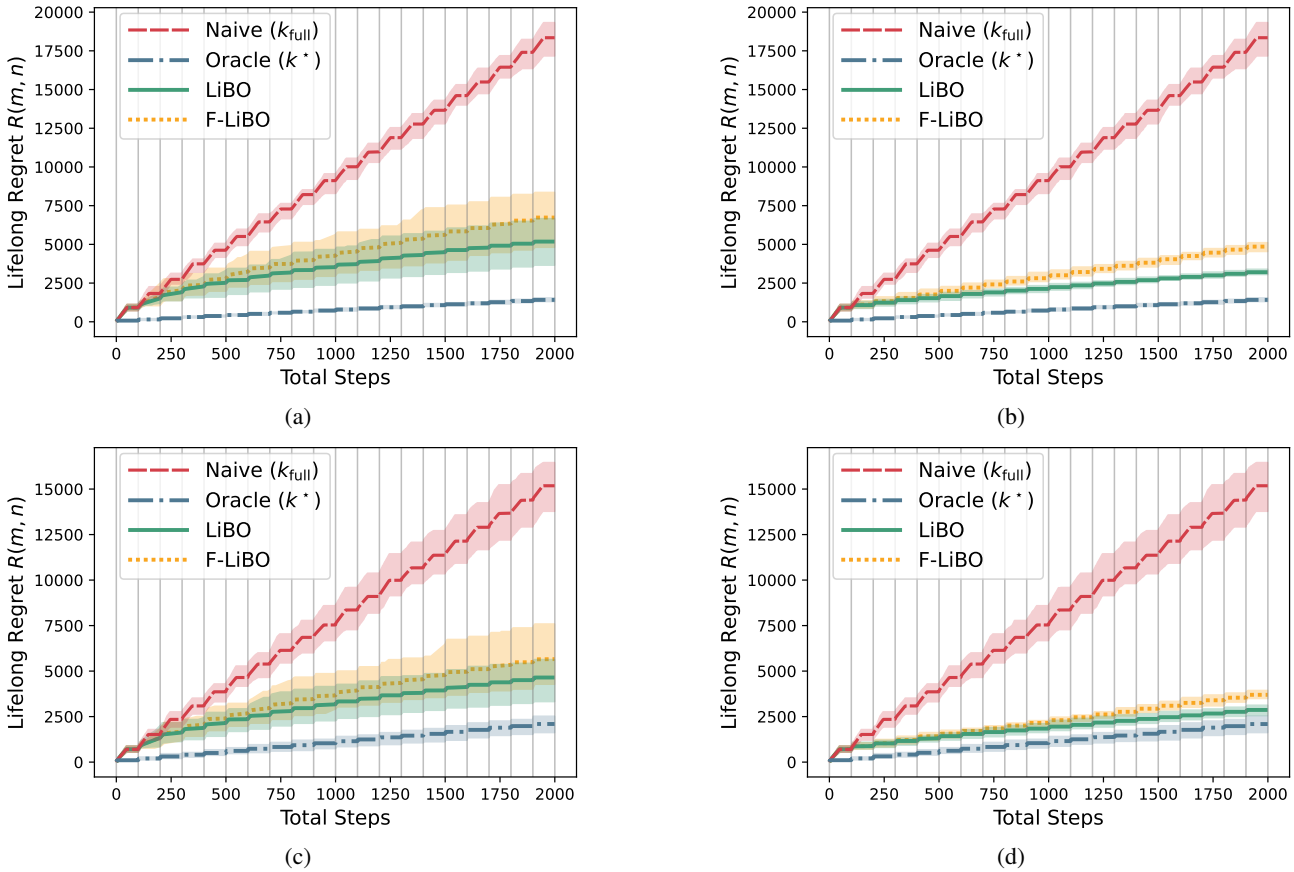

Figure 6: Lifelong regret with cosine basis (**a** & **b**) and Legendre polynomials as feature maps (**c** & **d**). In the plots **(a & c)** on the left, we use only the forced exploration data $\mathcal{D}_{1:s}^{\text{exp}}$ for meta-learning the kernel. For the plots **(b & d)** on the right, $D_{1:s}$, the data from all previous bandit interactions, is used. We observe that convergence is much faster when all interaction data is used in LiBO and F-LiBO.

only the forced exploration data is used for meta-learning. The plots on the right (i.e. Fig. b, d) correspond to a modified version of LiBO and F-LiBO where we use $D_{1:s}$, i.e., all previous bandit interactions, to meta-learn the kernel.

Generally, we observe that LiBO and F-LiBO substantially outperform the naive method which uses all base kernels. The gray vertical lines in Figure 6 indicate the beginning of a new task. We see that for every new task all algorithms initially experiences high regret, but, over time, as reward estimation improves, the cumulative regret flattens. As the rate of single-task convergence is dependent on the kernel, we see that differences in the performance between the algorithms emerge. When running LiBO, over time, forced exploration decreases and the estimated kernel converges to the true kernel. This means that, over time, the behavior of the agent using the LiBO estimator becomes indistinguishable form the agent using the oracle kernel. This is evident from 6 **(a)** as the slope of the single-task cumulative regret of the meta-agent (green) becomes the same as for the oracle agent (blue). In the federated case (yellow), while the estimated kernel also converges to the true kernel, the more restrictive setting forces us to use a constant exploration rate (see Algorithm 3 and Section 5) which means that the behavior of the federated meta-learner is always slightly sub-optimal. This can be observed by noting that the slope of the single-task cumulative regret of the federated meta-learner (yellow) is higher compared to the oracle agent even after the estimated kernel converges to the true kernel.

When we adjust LiBO and F-LiBO to use all available data to predict the kernel instead of only using $\mathcal{D}_{1:s}^{\text{exp}}$, the lifelong regret decreases. As we would expect, using more data for meta-learning the kernel speeds up the convergence of $\hat{k}$ to k^* which, in turn, makes the BO runs more efficient. In practice, using the data from all interactions, not just the ones obtained by forced exploration, seems to be the best choice. From a theoretical perspective, this comes with additional technical challenges, as we point out in Section 4.2.