(Private) Kernelized Bandits with Distributed Biased Feedback

FENGJIAO LI, Virginia Tech, USA
XINGYU ZHOU, Wayne State University, USA
BO JI, Virginia Tech, USA

In this paper, we study kernelized bandits with distributed biased feedback. This problem is motivated by several real-world applications (such as dynamic pricing, cellular network configuration, and policy making), where users from a large population contribute to the reward of the action chosen by a central entity, but it is difficult to collect feedback from all users. Instead, only biased feedback (due to user heterogeneity) from a subset of users may be available. In addition to such partial biased feedback, we are also faced with two practical challenges due to communication cost and computation complexity. To tackle these challenges, we carefully design a new *distributed phase-then-batch-based elimination (*DPBE*)* algorithm, which samples users in phases for collecting feedback to reduce the bias and employs *maximum variance reduction* to select actions in batches within each phase. By properly choosing the phase length, the batch size, and the confidence width used for eliminating suboptimal actions, we show that DPBE achieves a sublinear regret of $\tilde{O}(T^{1-\alpha/2} + \sqrt{\gamma_T T})$, where $\alpha \in (0, 1)$ is the user-sampling parameter one can tune. Moreover, DPBE can significantly reduce both communication cost and computation complexity in distributed kernelized bandits, compared to some variants of the state-of-the-art algorithms (originally developed for standard kernelized bandits). Furthermore, by incorporating various *differential privacy* models (including the central, local, and shuffle models), we generalize DPBE to provide privacy guarantees for users participating in the distributed learning process. Finally, we conduct extensive simulations to validate our theoretical results and evaluate the empirical performance.

CCS Concepts: • **Computing methodologies** → **Machine learning algorithms**; • **Theory of computation** → **Online learning algorithms**; *Communication complexity*; *Complexity theory and logic*; • **Security and privacy** → *Privacy protections.*

Additional Key Words and Phrases: kernelized bandits, distributed feedback, bias, regret, communication cost, computation complexity, privacy

ACM Reference Format:
Fengjiao Li, Xingyu Zhou, and Bo Ji. 2023. (Private) Kernelized Bandits with Distributed Biased Feedback. *Proc. ACM Meas. Anal. Comput. Syst.* 7, 1, Article 5 (March 2023), 47 pages. https://doi.org/10.1145/3579318

1 INTRODUCTION

Bandit optimization is a popular online learning paradigm for sequential decision making and has been widely used in a wide variety of real-world applications, including hyperparameter tuning [24], recommendation systems [23], and dynamic pricing [29]. In such problems, each decision point (called an arm or action), if chosen, yields an unknown reward. The goal of the agent is to maximize the cumulative reward by making proper decisions sequentially. An important way to capture general (e.g., *non-linear* and even *non-convex*) unknown objective functions is to consider a smoothness condition specified by a small norm of a Reproducing Kernel Hilbert Space (RKHS) associated with a kernel function. This setup is often referred to as *kernelized bandits*.

Authors' addresses: Fengjiao Li, fengjiaoli@vt.edu, Virginia Tech, 2202 Kraft Dr., Blacksburg, VA, USA, 24060; Xingyu Zhou, Wayne State University, Detriot, MI, USA, xingyu.zhou@wayne.org; Bo Ji, Virginia Tech, Blacksburg, VA, USA, boji@vt.edu.

2476-1249/2023/3-ART5
https://doi.org/10.1145/3579318

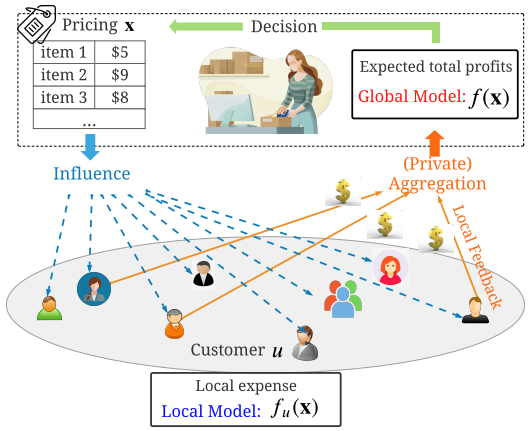


Fig. 1. Dynamic pricing: a motivating application of our problem.

Thanks to the strong link between RKHS functions and Gaussian processes (GP) [12, 19, 36], an extensive line of work has exploited GP models to estimate an unknown function f given a set of (noisy) evaluations of its values $f(\mathbf{x})$ at chosen actions $\mathbf{x}$. However, in many applications, the value $f(\mathbf{x})$ could represent an overall effect of action $\mathbf{x}$ on a large population of users where it is difficult for the learning agent to make direct observations; yet, the agent could collect some partial feedback from the distributed users in the population. In addition, feedback from these users could be biased due to user heterogeneity (e.g., different preferences). Therefore, we assume that each user u in the population is associated with a local function f_u, which is a function sampled from a GP with mean f. Consider the dynamic pricing problem [29] as an example (see Figure 1). When a company sets a pricing mechanism $\mathbf{x}$, this decision influences all the customers, and every customer, based on her individual demand and preference, makes a choice (purchase or not), which contributes to the total profits $f(\mathbf{x})$. Without knowing products' demand curves in advance, the company makes a sequence of pricing decisions with the goal of *maximizing profits while learning*. That is, the company aims to infer the expected demand and thus the expected profits f by collecting feedback from customers in each decision epoch. Note that it might be difficult for the company to collect feedback from *all* the customers - since purchases may take place at many local stores at different locations. For example, it is impractical for McDonald's headquarters to collect sales information from all of the nationwide customers within each decision epoch. Instead, the headquarter might be able to get feedback (i.e., sales information) from a subset of the customers. However, each customer's choice depends not only on her own preference towards the products and their prices but also on several other factors (location, competitors, promotion events, etc.), which is often *biased* feedback for the overall profits.

To that end, we study a new kernelized bandit setting where the agent could not get direct evaluations of the unknown reward function but only distributed biased feedback. We refer to this setting as *kernelized bandits with distributed biased feedback*. This bandit problem is shared by several other practical applications, including cellular network configuration [27] and public policy making [5]. However, existing learning algorithms developed for standard kernelized/GP bandits (e.g., GP-UCB [12, 36]) rarely consider such partial biased feedback in a distributed setting. To solve this new problem, a learning algorithm needs to be able to learn the unknown function from such biased feedback in a *sample-efficient* manner. Moreover, two practical challenges naturally arise in our problem: *communication cost* due to distributed learning [9] and *computation complexity* due to

GP update [7]. Therefore, not only need the learning algorithms be sample-efficient, but they must also be scalable in terms of both communication efficiency and computation complexity.

To that end, we propose the *learning with communication* framework where the biased feedback is communicated in phases, and design a new *distributed phase-then-batch-based elimination* algorithm that aggregates the distributed biased feedback in a communication-efficient manner and eliminates suboptimal actions in a computation-efficient manner while achieving a sublinear regret. Our main contributions are summarized as follows.

- To the best of our knowledge, this is the first work that studies a new kernelized bandit setting with distributed biased feedback, where three key challenges (user heterogeneity, communication efficiency, and computation complexity) inherently arise in the design of sample-efficient, scalable learning algorithms. While it is natural to consider phased elimination type of algorithms in such settings, the standard phased elimination algorithm relies on the so-called (near-)optimal experimental design [21], which cannot be directly applied to kernelized bandits due to the possible infinite feature dimension of RKHS functions.
- To that end, we design a new phased elimination algorithm, called *distributed phase-then-batch-based elimination (*DPBE*)*, which is carefully crafted to address all the aforementioned challenges. In particular, DPBE adds a *user-sampling* process to reduce the impact of bias from each individual user and selects actions according to *maximum variance reduction* within each phase. Moreover, a *batching* strategy is employed to improve both communication efficiency and computation complexity. That is, instead of selecting a new action at each round, DPBE plays the same action for a batch of rounds before switching to the next one. Not only does it help reduce the number of times one needs to compute the next action via GP update, but it also allows for reducing the dimensions of the vectors and matrices involved in both communication and computation.
- We show that DPBE achieves a sublinear regret of $\tilde{O}(T^{1-\alpha/2} + \sqrt{\gamma_T T})$[1] while incurring a communication cost of $O(\gamma_T T^{\alpha})$ and a computation complexity of $O((|\mathcal{D}|\gamma_T^3 + \gamma_T^4)\log T + \gamma_T T^{\alpha})$, where γ_T is the *maximum information gain* associated with the kernel of the unknown function f, $\mathcal{D}$ is the decision set, and $\alpha > 0$ is a user-sampling parameter that we can tune. It is worth noting that DPBE with $\alpha \in (0, 1)$ has a better computation complexity than some variants of the state-of-the-art algorithms (originally developed for standard kernelized bandits without biased feedback). Specifically, DPBE achieves three significant improvements compared to the state-of-the-art algorithms: (i) user-sampling efficiency ($O(T^{\alpha})$ vs. T), (ii) communication cost ($O(\gamma_T T^{\alpha})$ vs. T), and (iii) computation complexity ($O(\gamma_T T^{\alpha})$ vs. $O(T^3)$). Furthermore, we conduct extensive simulations to validate our theoretical results and evaluate the empirical performance in terms of regret, communication cost, and running time.
- Finally, we generalize our phase-then-batch framework to incorporate various *differential privacy* (DP) models (including the central, local, and shuffle models) into DPBE, which ensures privacy guarantees for users participating in the distributed learning process.

2 RELATED WORK

Kernelized bandits. Since [36] studied GP in the bandit setting, kernelized bandits (also called GP bandits) have been widely adopted to address black-box function optimization over a large or infinite domain [12]. Considering different application scenarios, kernelized bandits under different settings have recently been studied, including heavy-tailed payoffs [32], model misspecification [3], and corrupted rewards [4]. As typically considered in the literature, these works also assume that direct (noisy) feedback of the unknown function at a chosen action is available to the agent. In sharp

[1]The notation $\tilde{O}(\cdot)$ ignores polylog terms. Bounds on γ_T of different kernel functions can be found in Appendix A.2.

contrast, we study a new, practical setting where only distributed biased feedback can be obtained. Under this setting, not only does one need to use biased feedback in a sample-efficient manner, but one also has to consider communication efficiency, which is a common issue in distributed bandit-learning settings.

Distributed/collaborative kernelized bandits. While distributed or collaborative kernelized bandits have been studied recently [13, 14, 35], we highlight the key difference between our model and theirs as follows: motivated by real-world applications, we aim to learn one (global) bandit while most of them also aim to learn every local model, which results in quite different regret definitions (their group regret vs. our standard regret defined in Section 3). Moreover, they assume that every party (corresponding to a user in our problem) shares the same objective function. While [13] also studies similar bandit optimization with biased feedback, they assume a fixed number of local agents and bound the regret in terms of the distance between the target function and local functions, which could be very large. In addition, [13] does not consider communication efficiency, which is a key challenge in distributed learning.

Recently, the work of [22] studies a similar global reward maximization problem without direct feedback and also employs a phase-based elimination algorithm. However, the main difference is that they only consider linear bandits by assuming a linear reward function while we study kernelized bandits that can capture general *non-linear* and even *non-convex* functions and recover linear bandits as a special case when choosing a linear kernel. This strict generalization introduces three unique challenges: (i) different from the linearly parameterized bandits where the bias in the feedback can be quantified with a same-dimension random vector (i.e., $\xi_u = \theta_u - \theta^* \in R^d$ at each user u), it is unclear how to make an assumption of the bias in the non-parametric kernelized bandits setting in order to learn the unknown global reward function; (ii) due to the possible infinite feature dimension of functions in an RKHS, the (near-)optimal experimental design approach used in the phased-elimination algorithm for linear bandits cannot be directly adapted to kernelized bandits. Despite some recent efforts towards extending this experimental design based approach to kernelized bandits [8, 43], there still remain some key limitations (see our discussion below); (iii) since computation complexity is a critical bottleneck in kernelized bandits, a proper computation-efficient learning algorithm is desired when addressing our problem.

Experimental design for kernelized bandits. In [43], the authors propose to adaptively embed the feature representation of each action into a lower-dimensional space in order to apply the (near-)optimal experimental design for finite-dimensional actions. However, the intermediate regret due to the approximation error over T rounds is not considered at all because their goal is to find an ε-optimal arm at the end of T (i.e., a pure exploration problem) rather than minimizing the cumulative regret. While [8] aims at minimizing the cumulative regret, their algorithm and analysis are more complex than ours: it requires a non-standard robust estimator, obtaining an optimal distribution on the simplex, drawing samples from this distribution, and solving a second optimization problem. In contrast, we simply use the standard GP posterior mean and variance estimators, which can be computed in closed-form. Moreover, our algorithm can also be easily extended to handle infinite action sets (see Remark 4.2) rather than a finite set considered in [8].

3 PRELIMINARIES

Notation. Throughout this paper, we use lower-case letters (e.g., x) for scalars, lower-case bold letters (e.g., $\mathbf{x}$) for vectors, and upper-case bold letters (e.g., $\mathbf{X}$) for matrices. Let $[n] \triangleq \{1, \ldots, n\}$ denote any positive integer up to n, let $|\mathcal{U}|$ denote the cardinality of set $\mathcal{U}$, and let $\|\mathbf{x}\|_2$ denote the ℓ_2-norm of vector $\mathbf{x}$.

3.1 Problem Setting

We introduce a new kernelized bandit problem where the unknown function represents the overall reward over a large population containing an infinite number of users. The unknown reward function $f : \mathcal{D} \to \mathbb{R}$ is assumed to be fixed over a finite set of decisions $\mathcal{D} \subseteq \mathbb{R}^d$. At round t, the agent chooses an action $\mathbf{x}_t \in \mathcal{D}$, leading to a reward with mean $f(\mathbf{x}_t)$. This reward is unknown to the agent but captures the overall effectiveness of action $\mathbf{x}_t$ over the entire population $\mathcal{U}$, thus called *global reward*. Meanwhile, each user u in the population observes a (noisy) *local reward*: $y_{u,t} = f_u(\mathbf{x}_t) + \eta_{u,t}$ with mean $f_u(\mathbf{x}_t)$, where $\eta_{u,t}$ is the noise, and $f_u : \mathcal{D} \to \mathbb{R}$ is the local reward function, assumed to be an (unknown) realization of a random function (specified soon) with mean f. In this setting, the exact global reward corresponding to the entire population cannot be observed; only biased local reward feedback is available to the agent. We make the following assumptions about the unknown function f, the local function f_u, and the noise in the reward observations.

Assumption 1. We assume that function f is in the Reproducing Kernel Hilbert Spaces (RKHS), denoted by $\mathcal{H}_k$. Note that RKHS $\mathcal{H}_k$ is completely specified by its kernel function $k(\cdot, \cdot)$ (and vice-versa), with an inner product $\langle \cdot, \cdot \rangle_k$ obeying the reproducing property: $f(\mathbf{x}) = \langle f(\cdot), k(\mathbf{x}, \cdot) \rangle_k$ for all $f \in \mathcal{H}_k$ [12]. We list the most commonly used kernel functions (such as Squared Exponential (SE) and Matérn kernels) in Appendix A. Moreover, we assume that function f has a bounded norm: $\|f\|_k \triangleq \sqrt{\langle f, f \rangle_k} \leqslant B$, and that the kernel function is also bounded: $k(\mathbf{x}, \mathbf{x}) \leqslant \kappa^2$ for every $\mathbf{x} \in \mathcal{D}$, where both B and κ are positive constants.

Assumption 2. When the agent samples a user u to collect feedback, the local reward function f_u at u is assumed to be a function sampled from the GP with mean f and covariance[2] $k(\cdot, \cdot)$, i.e., $f_u \sim \mathcal{GP}(f(\cdot), k(\cdot, \cdot))$. In addition, we assume that each user is sampled independently for collecting feedback.

Assumption 3. We assume that the observation noise $\eta_{u,t} \sim \mathcal{N}(0, \sigma^2)$ is Gaussian with variance $\sigma > 0$ and that it is independent and identically distributed (*i.i.d.*) over time and across users.

The goal of the agent is to maximize the cumulative global reward, or equivalently, to minimize the regret defined as follows:

$$R(T) \triangleq \sum_{t=1}^{T} \left(\max_{\mathbf{x} \in \mathcal{D}} f(\mathbf{x}) - f(\mathbf{x}_t) \right). \tag{1}$$

3.2 Learning with Communication

For black-box function optimization based on noisy bandit feedback, kernelized bandit algorithms have shown strong empirical and theoretical performance. However, the agent in our problem setting does not have access to unbiased feedback of the object function f but has to collect biased feedback from distributed users from a large population. This scenario leads to the following framework of *learning with communication*.

Communication happens when some users are selected to report their feedback to the agent based on their biased local reward observations. By aggregating such biased feedback from the users, the agent improves her confidence in estimating function f and adjusts her decisions in the following rounds accordingly. To account for scalability, the agent collects distributed feedback from users periodically instead of immediately after making each decision. We call the time duration

[2]Our theoretical framework is applicable to a more general setting where the covariance of the local reward function is $v^2 k(\cdot, \cdot)$, i.e., $f_u \sim \mathcal{GP}(f(\cdot), v^2 k(\cdot, \cdot))$. This scaling parameter v^2 captures the variance of the bias in the local reward function f_u with its mean being the global reward function f. For this more general setting, our theoretical results still hold with only a slight adjustment to the posterior variance in the confidence width function (12).

between two communications as a *phase*. Consider a particular phase l. Let $\mathcal{T}_l$ be the set of round indices in the l-th phase and U_l be the set of selected users, called *participants*, that will report their feedback. With the actions $\{\mathbf{x}_t : t \in \mathcal{T}_l\}$ chosen by the agent in this phase, each user u in U_l sends the feedback $g(\{y_{u,t}\}_{t\in\mathcal{T}_l})$ to the agent at the end of the phase, where $g(\cdot)$ is a function (e.g., the average) of the local reward observations and is assumed to be the same for all users. Then, by aggregating all feedback $\{g(\{y_{u,t}\}_{t\in\mathcal{T}_l})\}_{u\in U_l}$, the agent estimates f and decides $\mathbf{x}_t$ for round t in the next phase $\mathcal{T}_{l+1}$. This learning with communication process is repeated until the end of T, with the goal of maximizing the cumulative (global) reward.

In this framework, we assume that the agent can employ some existing incentive mechanisms [26] in order to collect enough feedback for learning, but the cost has to be considered, e.g., the communication resources consumed for collecting feedback data. In addition, communication cost is also a critical factor in a general distributed learning system. In this work, we use the total quantity of communicated numbers (between the agent and all users) as another metric, in addition to the regret metric, to evaluate the communication efficiency of learning algorithms for our problem. Let L be the total number of phases in T rounds and $N_{u,l} \triangleq \dim\left(g(\{y_{u,t}\}_{t\in\mathcal{T}_l})\right)$ be the dimension of user u's feedback (which is the number of scalars in user u's feedback). Then, the total communication cost, denoted by $C(T)$, is as follows:

$$C(T) \triangleq \sum_{l=1}^{L} \sum_{u\in U_l} N_{u,l}. \tag{2}$$

In the following, we explain the learning with GP framework for standard kernelized bandits.

3.3 Learning with Gaussian Process

A Gaussian process (GP) over input domain $\mathcal{D}$, denoted by $\mathcal{GP}(\mu(\cdot), k(\cdot,\cdot))$, is a collection of random variables $\{f(\mathbf{x})\}_{\mathbf{x}\in\mathcal{D}}$ where every finite number of them $\{f(\mathbf{x}_i)\}_{i=1}^{n}, n \in \mathbb{N}$, is jointly Gaussian with mean $\mathbb{E}[f(\mathbf{x}_i)] = \mu(\mathbf{x}_i)$ and covariance $\mathbb{E}[(f(\mathbf{x}_i) - \mu(\mathbf{x}_i))(f(\mathbf{x}_j) - \mu(\mathbf{x}_j))] = k(\mathbf{x}_i, \mathbf{x}_j)$ for every $1 \leqslant i, j \leqslant n$. Hence, $\mathcal{GP}(\mu(\cdot), k(\cdot,\cdot))$ is specified by its mean function μ and a (bounded) covariance function $k : \mathcal{D}\times\mathcal{D} \to [0, \kappa^2]$. Assume that choosing action $\mathbf{x}_t$ at round t reveals a noisy observation:

$$y_t = f(\mathbf{x}_t) + \eta_t, \tag{3}$$

where $\eta_t \sim \mathcal{N}(0, \lambda)$ is a zero-mean Gaussian noise with variance $\lambda > 0$. Standard GP algorithms implicitly use $\mathcal{GP}(0, k(\cdot,\cdot))$ as the prior distribution over f. Then, given the observations $\mathbf{y}_t = [y_1, \ldots, y_t]^\top$ corresponding to a sequence of actions $\mathbf{X}_t = [\mathbf{x}_1^\top, \ldots, \mathbf{x}_t^\top]^\top$, the posterior distribution is also Gaussian with the mean and variance in the following closed-form:

$$\mu_t(\mathbf{x}) \triangleq \mathbf{k}(\mathbf{x}, \mathbf{X}_t)^\top (\mathbf{K}_{\mathbf{X}_t\mathbf{X}_t} + \lambda\mathbf{I})^{-1}\mathbf{y}_t, \tag{4}$$

$$\sigma_t^2(\mathbf{x}) \triangleq \mathbf{k}(\mathbf{x}, \mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{X}_t)^\top (\mathbf{K}_{\mathbf{X}_t\mathbf{X}_t} + \lambda\mathbf{I})^{-1}\mathbf{k}(\mathbf{x}, \mathbf{X}_t), \tag{5}$$

where $\mathbf{k}(\mathbf{x}, \mathbf{X}_t) = [k(\mathbf{x}, \mathbf{x}_s)]_{s=1,\cdots,t}^\top \in \mathbb{R}^{t\times 1}$ and $\mathbf{K}_{\mathbf{X}_t\mathbf{X}_t} = [k(\mathbf{x}, \mathbf{x}')]_{\mathbf{x},\mathbf{x}'\in\mathbf{X}_t} \in \mathbb{R}^{t\times t}$ is the corresponding kernel matrix.

Next, we introduce an important kernel-dependent quantity called *maximum information gain* [36]:

$$\gamma_t(k, \mathcal{D}) \triangleq \max_{\mathbf{X}\subseteq\mathcal{D}:|\mathbf{X}|=t} \frac{1}{2} \log\det\left(\mathbf{I} + \lambda^{-1}\mathbf{K}_{\mathbf{X}\mathbf{X}}\right), \tag{6}$$

which is often used to derive regret bounds. In addition, we have that $\gamma_t(k, \mathcal{D})$ scales sublinearly with t for most commonly used kernels (see Appendix A). For ease of notation, we often simply use γ_t to denote $\gamma_t(k, \mathcal{D})$ when the kernel function k and the dataset $\mathcal{D}$ are clear from the context.

Thanks to the strong connection between RKHS functions and GP [19] with the same kernel function k, one can use the above GP model to approximate unknown function $f \in \mathcal{H}_k$ within a reliable confidence interval with high probability.

4 ALGORITHM DESIGN

4.1 New Challenges and Main Ideas

In Section 3, we describe the learning with communication framework, which requires the distributed biased feedback to be communicated in phases and exhibits experimental scalability. This framework naturally leads us to consider a phased elimination algorithm that gradually eliminates suboptimal actions by periodically aggregating and analyzing the local feedback from the participants. However, several new challenges arise in our setting compared to the standard phase elimination algorithm in linear bandits [20, 21].

(i) How to select actions for each phase? The standard phase elimination algorithm often relies on the so-called near-optimal experimental design (i.e., a probability distribution over the currently active set) that minimizes the worst-case variance [20]. However, due to the possible infinite feature dimension of RKHS functions, adapting this approach to kernelized bandits setting is nontrivial even with the strong assumptions, requirements, and complicated algorithm design (e.g., [43] and [8], see discussion in Section 2). We are wondering if there is a simple and efficient method of selecting actions in each phase for our kernelized bandits setting. (Challenge ⓐ).

(ii) How to use biased feedback? In contrast to the standard phase elimination algorithm where feedback is unbiased, in our setting the local feedback from a particular user is biased. In order to reduce the impact of bias, an efficient user-sampling scheme is needed. However, how to incorporate this idea into the phase elimination algorithm is unclear (Challenge ⓑ).

(iii) How to deal with scalability? In our setting, scalability refers to both computation complexity and communication cost. On the one hand, it is well-known that standard GP bandits suffer a poor computation complexity (e.g., $O(T^3)$) due to the matrix inverse at each step for GP posterior update. On the other hand, due to the communication between the agent and the users, it is imperative to ensure a low communication cost (Challenge ⓒ).

Our approach. We propose a novel phase elimination algorithm that is able to simultaneously address all the above challenges. We highlight the main ideas as follows. (i) *User-sampling* for distributed biased feedback. In each phase, a well-tuned subset of users is sampled to reduce the impact of bias from each individual user. (ii) *Maximum variance reduction* for action selection. Upon selecting the next action within each phase, it simply selects the one that has the largest posterior variance. (iii) *Batching strategy* for scalability. Instead of selecting a new action at each round within a phase, it consistently plays the same action for a batch of rounds before selecting the next one, i.e., *rare-switching*. By reducing the number of times selecting a new action (which could be much smaller than the phase length), it also reduces the number of unique actions chosen within each phase, which can be utilized to improve the scalability in terms of both computation and communication through a proper design. Specifically, (a) *Computation*: via a *posterior reformulation* (specified in Section 4.2), we convert the dimension of the matrix in the inverse operation from the total rounds to the number of batches in each phase; (b) *Communication*: we let each participant *merge the local reward observations* in each batch before sending her feedback at the end of each phase. That is, the feedback $g(\{y_{u,t}\}_{t\in\mathcal{T}_l})$ from each participating user u in phase l is a vector, where each element corresponds to the average local reward of a batch. Then, the dimension of the feedback $g(\{y_{u,t}\}_{t\in\mathcal{T}_l}$ becomes the number of batches. For example, consider a particular phase with a total of 10 rounds. Without batching strategy, one requires to select an action for each round, i.e., 10 actions for this phase. However, the batching strategy selects an action for each batch. If

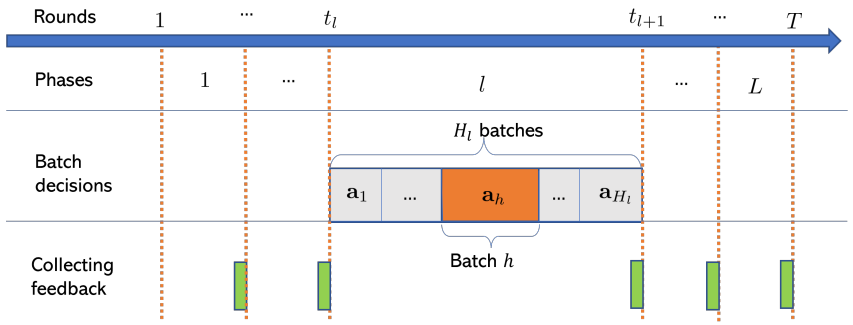


Fig. 2. The phase-then-batch strategy: T rounds are divided into L phases; at the end of each phase, participants report their feedback, which is used for deciding actions in the next phase; within each phase l, decisions are made in a batched fashion, e.g., playing $\mathbf{a}_h$ at all the rounds in the h-th batch.

each batch has size two, there are 5 batches in this phase, and the dimension of the matrix in the inverse operation is shrunk from 10 to 5, which will reduce the computation complexity about $10^3/5^3 = 8$ times for matrix inverse operations! In addition, by merging local observations of each unique action, only 5, instead of 10, (averaged) local rewards are communicated at each user.

4.2 Distributed Phase-then-Batch-based Elimination (DPBE)

Following the main ideas stated in the above section, we propose the phase-then-batch schedule strategy, shown in Figure 2 and design the distributed phase-then-batch-based elimination (DPBE) algorithm in Algorithm 1.

The DPBE algorithm is a phased elimination algorithm, which maintains a set $\mathcal{D}_l$ of active actions that are possible to be optimal and updates the active set after aggregating the distributed feedback.

Consider a particular phase l, DPBE has three main steps: 1) action selection (Lines 5-10); 2) distributed feedback collection (Lines 12-16); and 3) action elimination (Lines 17-21).

Before describing the details of DPBE, we explain some additional notations used in the algorithm. Throughout this paper, we use another notation "$\mathbf{a}$" to denote the specific chosen action under our algorithm to avoid too many subscripts or superscripts for all the batch, phase, or round indices. Consider the l-th phase. Let t_l and T_l be the time index right before the l-th phase and the length of the l-th phase, respectively. Then, the round indices in the l-th phase can be represented as $\mathcal{T}_l = \{t \in [T] : t_l + 1 \leqslant t \leqslant t_l + T_l\}$. In addition, $\mathcal{T}_l(\mathbf{a}) \triangleq \{t \in \mathcal{T}_l : \mathbf{x}_t = \mathbf{a}\}$ denotes the time indices when action $\mathbf{a}$ is selected in this phase, and H_l represents the number of batches in the l-th phase.

1) Action selection (Lines 5-10): In the l-th phase, actions are selected from the active set $\mathcal{D}_l$. As mentioned before, each selection is based on *maximum variance reduction* [39], and we employ batch schedule for scalability. Specifically, in the h-th batch, we find the action $\mathbf{a}_h$ that maximizes a reformulated posterior variance $\Sigma_{h-1}(\cdot)$ defined in Eq. (7) after $h-1$ batches (Eq. (8)). This is possible because the posterior variance can be computed without knowing any reward observations (see Eq. (5)). Then, play this action for $T_l(\mathbf{a}_h) \triangleq \lfloor (C^2-1)/\Sigma^2_{h-1}(\mathbf{a}_h) \rfloor$ rounds, which forms the h-th batch. Here, the batch size schedule is inspired by the *rare-switching* idea in [1, 7]. This batch schedule strategy enables us to merge rounds and thus shrink the dimensions of the matrix and vectors used for computing the variance in Eq. (5). By the end of each batch, we update the variance function by incorporating the action in the current batch. Let $\mathbf{A}_h = [\mathbf{a}_1^\top, \ldots, \mathbf{a}_h^\top]^\top \in \mathbb{R}^{h \times d}$ be the $h \times d$ matrix that contains the h chosen actions so far. We reformulate the standard posterior variance in Eq. (5)

Algorithm 1 Distributed Phase-then-Batch-based Elimination (DPBE)

1: **Input:** $\mathcal{D} \subseteq \mathbb{R}^d$, parameters $\alpha > 0$, $\beta \in (0,1)$, C, and local noise σ^2
2: **Initialization:** $l = 1$, $\mathcal{D}_1 = \mathcal{D}$, $t_1 = 0$, and $T_1 = 1$
3: **while** $t_l < T$ **do**
4: Set $\tau = 1$, $h = 0$, $\tau_1 = 0$ and $\Sigma_0^2(\mathbf{x}) = k(\mathbf{x}, \mathbf{x})$, for all $\mathbf{x} \in \mathcal{D}_l$
5: **while** $\tau \leqslant T_l$ **do**
6: $h = h + 1$
7: Choose action

$$\mathbf{a}_h \in \operatorname*{argmax}_{\mathbf{x} \in \mathcal{D}_l} \Sigma_{h-1}^2(\mathbf{x}) \tag{8}$$

8: Play action $\mathbf{a}_h$ for $T_l(\mathbf{a}_h) \triangleq \lfloor (C^2 - 1)/\Sigma_{h-1}^2(\mathbf{a}_h) \rfloor$ rounds if not reaching $\min\{T, t_l + T_l\}$
9: Update $\tau = \tau + T_l(\mathbf{a}_h)$, and incorporate $\mathbf{a}_h$ into the posterior variance $\Sigma_h^2(\cdot)$ (see Eq. (7))
10: **end while**
11: Let $H_l = h$ denote the total number of batches in this phase
12: Randomly select $\lceil 2^{\alpha l} \rceil$ participants U_l
Operations at each participant
13: **for** each participant $u \in U_l$ **do**
14: Collect and compute local average reward for every chosen action $\mathbf{a} \in \mathbf{A}_{H_l}$:

$$y_l^u(\mathbf{a}) = \frac{1}{T_l(\mathbf{a})} \sum_{t \in \mathcal{T}_l(\mathbf{a})} y_{u,t}$$

15: Send the (local) average reward for each chosen action $\mathbf{y}_l^u \triangleq [y_l^u(\mathbf{a})]_{\mathbf{a} \in \mathbf{A}_{H_l}}$ to the agent
16: **end for**
17: Aggregate local observations for each chosen action $\mathbf{a} \in \mathbf{A}_{H_l}$:

$$y_l(\mathbf{a}) = \frac{1}{|U_l|} \sum_{u \in U_l} y_l^u(\mathbf{a}) \tag{9}$$

18: Let $\bar{\mathbf{y}}_l = [y_l(\mathbf{a}_1), \ldots, y_l(\mathbf{a}_{H_l})]$
19: Update $\bar{\mu}_l(\cdot)$ according to Eq. (11)
20: Eliminate low-rewarding actions from $\mathcal{D}_l$ based on the confidence width $w_l(\cdot)$ in Eq. (12):

$$\mathcal{D}_{l+1} = \left\{ \mathbf{x} \in \mathcal{D}_l : \bar{\mu}_l(\mathbf{x}) + w_l(\mathbf{x}) \geqslant \max_{\mathbf{b} \in \mathcal{D}_l} (\bar{\mu}_l(\mathbf{b}) - w_l(\mathbf{b})) \right\} \tag{10}$$

21: $T_{l+1} = 2T_l$, $t = t + T_l$, $l = l + 1$
22: **end while**

and update the posterior variance as follows:

$$\Sigma_h^2(\mathbf{x}) \triangleq k(\mathbf{x}, \mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{A}_h)^\top (\mathbf{K}_{\mathbf{A}_h \mathbf{A}_h} + \lambda \mathbf{W}_h^{-1})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{A}_h), \tag{7}$$

where $\mathbf{W}_h \in \mathbb{R}^{h \times h}$ is a diagonal matrix with $[W_h]_{ii} = T_l(\mathbf{a}_i)$ for any $i \in [h]$, and λ is set to be the noise variance of local observations, i.e., $\lambda = \sigma^2$. Here, we reformulate the standard posterior variance in Eq. (5) with Eq. (7) in order to save computation complexity (especially for computing matrix inverse) while maintaining the same order of regret (sacrificing only a constant multiplier).

2) Distributed feedback collection (Lines 12-16): To reduce the impact of bias from some specific user(s), the agent randomly samples a subset of users (called participants) U_l from $\mathcal{U}$ to participate in the learning process (Line 12). We let $|U_l| = \lceil 2^{\alpha l} \rceil$, where the user-sampling

parameter $\alpha > 0$ is an input of the algorithm. Recall that H_l denotes the number of batches in the l-th phase. Each participant $u \in U_l$ collects their local reward observations of each chosen action $\mathbf{a} \in \mathbf{A}_{H_l}$ and send the average $y_l^u(\mathbf{a})$ for every chosen action $\mathbf{a} \in \mathbf{A}_{H_l}$ as feedback to the agent, i.e., $g(\{y_{u,t}\}_{t\in\mathcal{T}_l}) = \mathbf{y}_l^u \triangleq [y_l^u(\mathbf{a})]_{\mathbf{a}\in\mathbf{A}_{H_l}}$. Note that the dimension of the feedback depends on the number of batches, which is also the communication cost associated with each participant (Eq. (2)). Therefore, *by employing the idea of rare switching, we reduce both computation complexity and communication cost (ⓒ).*

3) Action elimination (Lines 17-21): Aggregate (specifically, average) the feedback from the participants for each action $\mathbf{a} \in \mathbf{A}_{H_l}$ (Line 17). Then, using the aggregated feedback (i.e., the averaged local reward $\bar{\mathbf{y}}_l = [y_l(\mathbf{a}_1), \ldots, y_l(\mathbf{a}_{H_l})]$ of the chosen actions $\mathbf{a} \in \mathbf{A}_{H_l}$), the agent can compute the posterior mean function reformulated as follows (Line 19):

$$\bar{\mu}_l(\mathbf{x}) \triangleq \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \bar{\mathbf{y}}_l. \tag{11}$$

Considering the bias in the feedback due to user heterogeneity (ⓑ), we carefully construct a confidence width $w_l(\cdot)$ that incorporates both the noise and bias as follows:

$$w_l(\mathbf{x}) \triangleq \sqrt{\frac{2k(\mathbf{x},\mathbf{x})\log(1/\beta)}{|U_l|}} + \sqrt{\frac{2\Sigma_{H_l}^2(\mathbf{x})\log(1/\beta)}{|U_l|}} + B\Sigma_{H_l}(\mathbf{x}), \tag{12}$$

where B is the bound of f's kernel norm, and β is the confidence level from the input. Using this confidence width $w_l(\cdot)$ and the mean estimator function $\bar{\mu}_l(\cdot)$ in Eq. (11), we can identify suboptimal actions with high probability (*w.h.p.*). Finally, we update the set of active actions $\mathcal{D}_{l+1}$ by eliminating the suboptimal actions from $\mathcal{D}_l$ (Line 20).

Remark 4.1 (Merge batches). *For implementation, we also merge different batches with the same chosen action in each phase. By doing this, we further shrink the dimension of the matrix in the inverse operation (thus reducing the time complexity) and also the dimension of local feedback (thus reducing the communication cost).*

Remark 4.2 (General decision set). *Following the techniques used in [25],* DPBE *can also be extended from a finite domain to a continuous domain (e.g., $\mathcal{D} = [0,1]^d$) via a simple discretization trick and Lipschitz continuity of functions under commonly used kernels.*

5 MAIN RESULTS

In this section, we present the performance of our proposed DPBE algorithm in terms of regret, computation complexity, and communication cost, respectively.

First, we analyze the regret performance of DPBE and present the upper bound in Theorem 5.1. While the DPBE algorithm uses GP tools to define and manage the uncertainty in estimating the unknown function f, the analysis of DPBE algorithm does not rely on any *Bayesian* assumption about f being drawn from the prior $\mathcal{GP}(0, k(\cdot,\cdot))$, and it only requires f to be bounded in the kernel norm associated with the RKHS $\mathcal{H}_k$.

Theorem 5.1 (Regret). *Let $\beta = \frac{1}{|\mathcal{D}|T}$. Under Assumptions 1, 2 and 3, the* DPBE *algorithm achieves the following expected regret:*

$$\mathbb{E}[R(T)] = O(T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)}) + O(\sqrt{\gamma_T T}) + O(\sqrt{\gamma_T T^{1-\alpha}\log(|\mathcal{D}|T)}). \tag{13}$$

We provide the detailed proof of Theorem 5.1 in Appendix C. Bounds for γ_T of different kernels can be found in Appendix A.2. In the following, we make two remarks about the above result.

Remark 5.2. *In the above regret upper bound (RHS of Eq. (13)), the first term, $O(T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)})$, is due to the bias in the feedback at heterogeneous participants, and the last two terms, $O(\sqrt{\gamma_T T})$*

+ $O(\sqrt{\gamma_T T^{1-\alpha}\log(|\mathcal{D}|T)})$, are from the noisy feedback of each action as in the standard kernelized bandits (cf. [36]). Note that the first term (i.e., the regret caused by the bias) can be improved if one increases the number of sampled users in the learning process (i.e., choosing a larger value of α). However, this would also result in a larger communication cost.

Remark 5.3 (Maximum Uncertainty Reduction). *Recall that* DPBE *selects actions that have maximum variance for each batch (Eq. (5)). Intuitively, variance at action $\mathbf{x}$ indicates the uncertainty about $f(\mathbf{x})$, and thus, maximum-variance selection leads to maximum uncertainty reduction, which promotes exploration.*

Remark 5.4 ((Sub-)optimality). *We first note that one natural lower bound for our setting is the one for the standard setting of kernelized bandits, where the agent receives unbiased feedback after taking an action. In this setting, the state-of-the-art lower bounds under two commonly-used kernel functions (SE and Matérn)*[3] *are summarized in Table 5 (see Appendix A.2), which can also serve as valid lower bounds for the setting we consider. Recall that $\alpha > 0$ is the user-sampling parameter that one can choose. We discuss the (sub-)optimality of our upper bounds in two cases: $\alpha \geq 1$ (i.e., the high-communication regime) and $\alpha \in (0, 1)$ (i.e., the low-communication regime). (i) In the high-communication regime, the upper bound in (13) now becomes $O(\sqrt{\gamma_T T})$, which is* near-optimal *under both SE and Matérn kernels. In particular, if one plugs the best-known bounds on γ_T for SE and Matérn kernels (as listed in the first column in Table 5; also see [40]) into the regret upper bound $O(\sqrt{\gamma_T T})$, one can now have explicit regret upper bounds (as listed in the third column in Table 5), which match the corresponding lower bounds, up to only a logarithmic factor. (ii) In the low-communication regime, the first term in the regret upper bound (see Eq. (13) in Theorem 5.1) that depends on α may be dominant and cannot be ignored. On the other hand, the existing lower bounds do not depend on α since they are derived under the standard setting of kernelized bandits, where user sampling is irrelevant. Therefore, an important open problem is to close the gap by deriving tighter lower and/or upper bounds that capture the effect of user sampling in the new setting with distributed biased feedback we consider. We leave it as our future work.*

As a critical bottleneck of kernelized bandits algorithms, the computation complexity of DPBE algorithm is analyzed in the following Theorem 5.5.

Theorem 5.5 (Computation complexity). *The computation complexity of* DPBE *is at most $O(\gamma_T T^{\alpha} + (|\mathcal{D}|\gamma_T^3 + \gamma_T^4)\log T)$.*

Proof. Recall that H_l is the number of batches in the l-th phase. Then, the computation complexity of the central agent in the l-th phase is upper bounded by the following:

$$O(H_l \cdot H_l^3 + H_l \cdot |\mathcal{D}_l|H_l^2 + |U_l|H_l + |\mathcal{D}_l|H_l^2).$$

Specifically, for each $h \in [H_l]$ within phase l, the agent would compute the matrix inverse in Eq. (7), the complexity of which is at most $O(h^3) \leq O(H_l^3)$. With this matrix inverse result ready, the agent can solve the maximum-variance problem in Eq. (8) with at most $O(|\mathcal{D}_l|H_l^2)$ for each batch and determine the batch length $\mathcal{T}_l(\mathbf{a}_h)$ with $O(1)$ after we have the posterior variance. Since there is a total of H_l batches for phase l, the total complexity up to this stage is $O(H_l \cdot H_l^3 + H_l \cdot |\mathcal{D}_l|H_l^2)$. Finally, in the elimination stage for phase l, the agent first loads/aggregates all the feedbacks with $O(|U_l|H_l)$ and can again reuse the matrix inverse result so that only $O(|\mathcal{D}_l|H_l^2)$ is required to eliminate all the bad arms.

[3]Note that even for the standard setting of kernelized bandits, there only exist lower bounds for these specific kernel functions rather than a general one in terms of the maximum information gain γ_T.

Table 1. Comparison of computation complexity under DPBE and three state-of-the-art algorithms.

Algorithms	Complexity
GP-UCB [12]	$O(\lvert\mathcal{D}\rvert T^3)$
BBKB [6]	$O(\lvert\mathcal{D}\rvert T\gamma_T^2)$
MINI-GP-Opt [7]	$O(T + \lvert\mathcal{D}\rvert\gamma_T^3 + \gamma_T^4)$
DPBE (this paper)	$O(\gamma_T T^\alpha + (\lvert\mathcal{D}\rvert\gamma_T^3 + \gamma_T^4)\log T)$

Putting the two stages together, we have the above result. Thus, it remains to bound the number of batches H_l within each phase l. Fortunately, inspired by [7], we are able to show that H_l can be upper bounded by the maximum information gain. We state this result in Lemma 5.6 and provide the proof in Appendix D.

LEMMA 5.6 (BOUND ON H_l). *For any phase l, the number of batches H_l is at most $\frac{4\sigma^2C^2}{C^2-1}\gamma_T$.*

We can get that the total number of phases is $O(\log T)$ and the total number of participants satisfies $O(T^\alpha)$. Armed with all the above results, we arrive at our final computation complexity. □

REMARK 5.7 (COMPLEXITY COMPARISON). *For comparison, we list the computation complexity of the state-of-the-art algorithms for standard kernelized bandits in Table 1. As we already know,* GP-UCB *has a computation complexity of $O(\lvert\mathcal{D}\rvert T^3)$, because it requires computing the posterior mean and variance using $O(T^2)$ and then finds the action that maximizes the UCB function per step. Recently, BBKB in [6] improves the time complexity to $(\lvert\mathcal{D}\rvert T\gamma_T^2)$, and later MINI-GP-Opt in [7] further reduces computation complexity to $O(T + \lvert\mathcal{D}\rvert\gamma_T^3 + \gamma_T^4)$, which is currently the fastest no-regret algorithm. Although more feedback is needed to address the additional bias in our setting, our algorithm can still achieve an improvement with the highest order term being $O(\gamma_T T^\alpha)$. This improvement comes from the fact that the participants help preprocess local reward observations before sending them out.*

Meanwhile, the bound on H_l also allows us to achieve a meaningful communication cost.

THEOREM 5.8 (COMMUNICATION COST). *DPBE incurs at most $O(\gamma_T T^\alpha)$ communication cost.*

The proof for Theorem 5.8 is also provided in Appendix D.

REMARK 5.9 (COMMUNICATION COST WHEN MERGING BATCHES). *By further merging batches according to Remark 4.1, the* DPBE *algorithm incurs $O(\min\{\gamma_T, \lvert\mathcal{D}\rvert\}T^\alpha)$ communication cost; We highlight that the batch schedule strategy plays a key role in obtaining the above bounds. Otherwise, even merging rounds as Remark 4.1 with the reformulated representation in Eqs. (7) and (11), the dimension of the local feedback at each participant is $O(\min\{T_l, \lvert\mathcal{D}_l\rvert\})$ in order to distinguish different actions, which leads to $O(\min\{T, \lvert\mathcal{D}\rvert\}T^\alpha)$ (vs. ours $O(\min\{\gamma_T, \lvert\mathcal{D}\rvert\}T^\alpha)$).*

6 DIFFERENTIALLY PRIVATE DPBE

As privacy is also an important factor in distributed learning, it is critical to protect users' sensitive data when collecting and aggregating their feedback. For example, in the dynamic pricing application, it is required that an adversary cannot infer a customer's private information (e.g., purchase or not) by observing the pricing mechanism set by the company. Moreover, users may require more stringent privacy protection in some applications — users are not willing to share their perceived Quality-of-Experience (QoE) directly with the central controller in the cellular network configuration problem; citizens are not willing to reveal the information about their preference for a certain policy to the government. Formally, we adopt the concept of *differential privacy* (DP) [17] as

the privacy metric. Thanks to the phase-then-batch schedule strategy in our algorithm, different DP trust models (e.g., central [17], local [42], and shuffle [11]) can be applied through proper designs. In this section, we describe how to ensure DP under DPBE with a trusted agent (the central DP model) and also analyze the regret under such a DP model. Extensions of the differentially private DPBE algorithms in other DP models (e.g., the stronger local DP model) are presented in Appendix E.

6.1 DP Definition and Algorithm

In the central DP model, we assume that each participating user trusts the agent, and hence, the agent can collect their raw data (i.e., the local reward $\mathbf{y}_l^u$ in our case). The privacy guarantee is that any adversary with arbitrary auxiliary information cannot infer a particular user's data by observing the decisions of the agent. To achieve this privacy protection, the central DP model requires that the decisions of the agent on two neighboring sets of users (differing in only one user) are indistinguishable [16]. Formally, we have the following definition.

Definition 6.1. (Differential Privacy (DP)). For any $\varepsilon \geqslant 0$ and $\delta \in [0, 1]$, a randomized algorithm $\mathcal{M}$ is (ε, δ)-differentially private (or (ε, δ)-DP) if for every pair of $U, U' \subseteq \mathcal{U}$ differing on a single participant and for any subset of output actions $\mathbf{Z} = [\mathbf{z}_1^\top, \ldots, \mathbf{z}_T^\top]^\top$, we have

$$\mathbb{P}[\mathcal{M}(U) = \mathbf{Z}] \leqslant e^{\varepsilon}\mathbb{P}[\mathcal{M}(U) = \mathbf{Z}] + \delta. \tag{14}$$

The parameters ε and δ indicate how private $\mathcal{M}$ is; the smaller, the more private. According to the post-processing property of DP (cf. Proposition 2.1 in Dwork et al. [17]), it suffices to guarantee that the aggregator (Line 17 in Algorithm 1) is (ε, δ)-DP. To achieve this, the standard Gaussian mechanism can be applied by adding Gaussian noise to the aggregated distributed feedback. Then, the *private* aggregated feedback for the chosen actions in the l-th phase becomes

$$\tilde{\mathbf{y}}_l = \bar{\mathbf{y}}_l + (\rho_1, \ldots, \rho_{H_l}), \tag{15}$$

where $\rho_j \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_{nc}^2)$ and the variance σ_{nc}^2 (specified in Eq. (66)) is based on the (high-probability) sensitivity of of the average vector $\bar{\mathbf{y}}_l$. In addition, we replace $\bar{\mathbf{y}}_l$ with $\tilde{\mathbf{y}}_l$ in Eq. (15) to obtain the private mean estimator:

$$\tilde{\mu}_l(\mathbf{x}) \triangleq \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \tilde{\mathbf{y}}_l. \tag{16}$$

The confidence width function is also updated by counting the uncertainty introduced by privacy noise as follows:

$$\tilde{w}_l(\mathbf{x}) \triangleq \sqrt{\frac{2k(\mathbf{x}, \mathbf{x}) \log(1/\beta)}{|U_l|}} + \sqrt{\frac{2\Sigma_{H_l}^2(\mathbf{x}) \log(1/\beta)}{|U_l|}} + B\Sigma_{H_l}(\mathbf{x}) + \sqrt{2\sigma_n^2 \log(1/\beta)}, \tag{17}$$

where σ_n is related to the overall privacy noise and will be specified in the algorithm. We present the differentially private version of DPBE, called DP-DPBE, in Algorithm 2 (see Appendix E).

6.2 Performance Guarantees

In the following, we provide the main results of the DP-DPBE algorithm in terms of privacy guarantee and regret. We start by stating an additional assumption in Assumption 4. This one-time participation assumption is commonly used in private bandits (see, e.g., [15, 28, 33, 38]). To handle multiple-times participation, one can use (adaptive) composition theorem of differential privacy [17].

Assumption 4. Each sampled user only participates in one phase of the learning process.

Then, we present the privacy guarantee in Theorem 6.2 and provide the proof in Appendix E.2.

Theorem 6.2 (Privacy Guarantee). *Under Assumptions 1, 2, 3, and 4, for any $\varepsilon > 0$ and $\delta \in (0,1)$, the* DP-DPBE *algorithm (Algorithm 2) guarantees (ε, δ)-DP.*

As an additional Gaussian noise is injected to protect privacy, DP-DPBE suffers additional regret cost. We present its regret upper bound in Theorem 6.3.

Theorem 6.3 (Regret of DP-DPBE). *Under Assumptions 1, 2, and 3, the* DP-DPBE *algorithm (Algorithm 2) with $\beta = \frac{1}{|\mathcal{D}|T}$ achieves the following expected regret:*

$$\mathbb{E}[R(T)] = O(T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)}) + O\left(\frac{\ln(1/\delta)\gamma_T T^{1-\alpha}\sqrt{\log(|\mathcal{D}|T)}}{\varepsilon}\right). \tag{18}$$

The full proof of Theorem 6.3 is provided in Appendix E.3. Regarding this regret result, we make the following remark.

Remark 6.4 (Privacy "for free"). *Comparing Theorem 6.3 with Theorem 5.1, we see that the additional regret cost introduced by privacy noise is $\tilde{O}\left(\frac{\ln(1/\delta)\gamma_T T^{1-\alpha}}{\varepsilon}\right)$, which is a lower order term compared to the first non-private term. This implies that our* DP-DPBE *algorithm enables us to achieve a privacy guarantee "for free" in the kernelized bandits setting. The same observation of achieving privacy "for free" is also observed in a recent study [22] that only considers linear bandits. However, our result is a strict generalization in the sense that it holds for general functions and recovers their result when considering a linear kernel.*

Remark 6.5 (Other DP models). *In the cases where the users do not trust the agent, users' data privacy has to be protected by the users themselves as in a local DP model or by resorting to a third party, e.g., the shuffler in the shuffle DP model. In Appendix E, we make extensions of* DP-DPBE *by considering these two trust models. In the local model, a local randomizer is equipped with each participant so that the feedback from each user is private. While the local model ensures a stronger privacy guarantee compared to the central DP, it always incurs a larger additional regret cost. Meanwhile, thanks to the phase-based strategy in* DPBE, DP-DPBE *can also be easily extended to the shuffle model, which achieves better regret-privacy tradeoff as in [22], i.e., achieving nearly the same regret as the central model, yet without the need to assume a trustworthy agent.*

7 NUMERICAL EXPERIMENTS

We now evaluate our proposed approach empirically on three types of functions: 1) synthetic functions in the RKHS with an SE kernel, 2) standard benchmark functions (with an unknown RKHS norm) [37] , and 3) functions from a real-world dataset. We implement the algorithms in python and run the numerical experiments on a Dell desktop (Processor: Intel®Core i7 CPU, 8 cores; Memory: 32GB).

7.1 Synthetic Function

We follow [18] to construct the global function f from the RKHS by sampling $m = 30d$ independent points, $\widehat{\mathbf{x}}_1, \ldots, \widehat{\mathbf{x}}_m$, uniformly on $[0,1]^d$, and $\widehat{a}_1, \ldots, \widehat{a}_m$, uniformly on $[-1,1]$, and defining $f(\mathbf{x}) = \sum_{i=1}^m \widehat{a}_i k(\widehat{\mathbf{x}}_i, \mathbf{x})$ for all $\mathbf{x} \in \mathcal{D}$, where k is SE kernel with length-scale $l_{SE} = 0.2$. The RKHS norm is $\|f\|_k^2 = \sum_{i=1}^m \sum_{j=1}^m \widehat{a}_i \widehat{a}_j k(\widehat{\mathbf{x}}_i, \widehat{\mathbf{x}}_j)$, which is assumed to be known. Each local reward function f_u, a random function sampled from a given Gaussian process, is generated by following Algorithm 1 in [19]. In the simulations, we evaluate the algorithms in a more general setting with $f_u \sim \mathcal{GP}(f(\cdot), v^2 k(\cdot,\cdot))$, where v^2 is a scaling parameter that can be used to set a reasonable level of local bias (see Footnote 2).

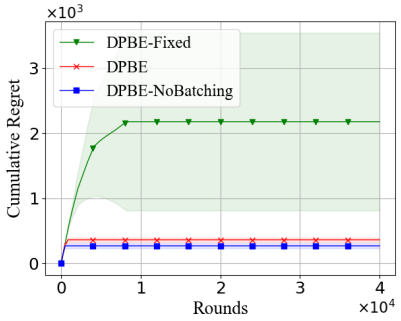


Fig. 3. Comparison of regret performance on a synthetic function. The shaded area represents the standard deviation

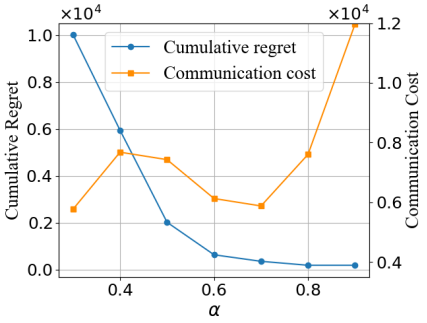


Fig. 4. The regret and communication cost under DPBE with different values of α.

Table 2. Comparisons of communication cost and running time under DPBE, DPBE-Fixed, and DPBE-NoBatching on a synthetic function.

Algorithms	Communication cost	Running time (seconds)
DPBE	5.87×10^3	0.12
DPBE-Fixed	5.87×10^3	0.19
DPBE-NoBatching	1.81×10^4	0.61

7.1.1 Ablation Studies and Analysis. First, we show that the DPBE algorithm that selects actions according to maximum variance reduction achieves sublinear regret, as shown in Figure 3. Then, we perform numerous ablation studies to confirm the efficacy of other two key components in our algorithm: user-sampling and batching strategy. To this end, we consider the corresponding variants of our algorithm. In this simulation, we perform 20 runs for each algorithm by setting $|\mathcal{D}| = 100, d = 3, C = 1.6, \sigma = 0.01, v = 0.1, T = 40000, \alpha = 0.7, \beta = 1/(|\mathcal{D}|T)$ and $\lambda = \sigma^2/v^2$ and present the regret performance in Figure 3 and communication cost and runtime in Table 2.

1) Importance of (exponentially-increasing) user-sampling. To this end, we consider the first variation of DPBE with a fixed number of participants, called DPBE-Fixed, where the number of participants in each phase is fixed at $|U| = \lfloor \frac{\sum_{l=1}^{L} |U_l| * N_{u,l}}{\sum_{l=1}^{L} N_{u,l}} \rfloor$ so as to have the same communication cost as DPBE. From Figure 3, we observe that DPBE with exponentially-increasing user-sampling over phases performs much better than DPBE-Fixed with the same communication cost. It demonstrates that the exponentially-increasing user-sampling mechanism in DPBE is critical to striking a balance between regret and communication cost. From Table 2, we observe that DPBE-Fixed takes a little longer time than DPBE. This is mainly because DPBE-Fixed needs more phases to find the optimal action (i.e., L is larger when $|\mathcal{D}_L| = 1$).

2) Benefits of batching strategy. To illustrate the impact of batching schedule strategy, we consider another variant of DPBE that does not employ batching strategy, called DPBE-NoBatching. In particular, it selects an action according to Eq. (8) for each round in any phase. Without batching strategy, DPBE-NoBatching communicates local observations directly without merging, and computes the posterior mean and variance according to standard update formula: Eq. (4) and Eq. (5) respectively; From Figure 3, we observe that DPBE, similar to other *rare-switching* algorithms [1], achieves a slightly worse regret performance than DPBE-NoBatching. However, as shown in

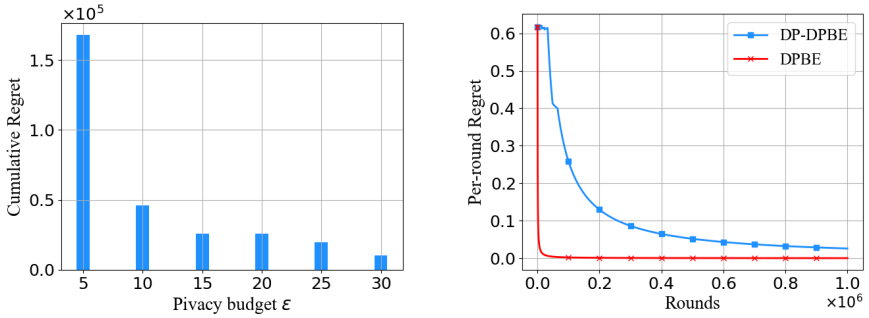


Fig. 5. Performance of DP-DPBE. (a) Final cumulative regret vs. the privacy budget ε with $\delta = 10^{-6}$; (b) Per-round regret vs. time with parameters $\varepsilon = 15$ and $\delta = 10^{-6}$.

Table 2, it significantly saves communication cost ($\sim 3\times$) by merging local observations in batches and reduces computation time ($\sim 5\times$) by shrinking the dimension of posterior reformulations.

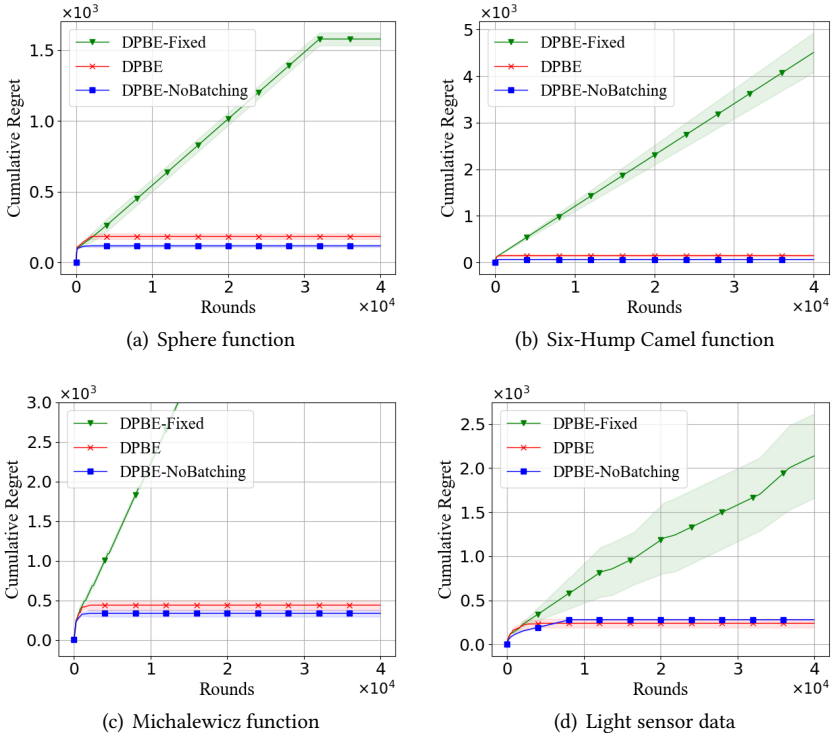
7.1.2 Regret-communication Tradeoff. We now turn to investigate the regret-communication tradeoff captured by the user-sampling parameter α, as shown in Theorem 5.1.

Consider $\alpha \in \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. The cumulative regret and total communication cost of DPBE with different values of α are presented in Figure 4. As expected, while a larger value of α yields a lower regret, it generally results in a higher communication cost. Notice that DPBE incurs slightly higher communication cost when $\alpha = \{0.4, 0.5, 0.6\}$ compared to $\alpha = 0.7$, this is mainly because DPBE with a smaller value of α needs more phases to find the optimal action (i.e., L is larger when $|\mathcal{D}_L| = 1$). One can tune the user-sampling parameter α to achieve a better regret-communication cost accordingly, e.g., $\alpha = 0.7$ for this synthetic function setting.

7.1.3 Regret-privacy Tradeoff. Finally, we evaluate the performance of the differentially private DPBE, i.e., DP-DPBE, and present the result in Figure 5. Figure 5(a) shows how the cumulative regret at the end of $T = 10^6$ rounds varies with different values of privacy parameters $\varepsilon \in \{5, 10, 15, 20, 25, 30\}$ and $\delta = o(1/T) = 10^{-6}$, which reveals a tradeoff between regret and privacy. Figures 5(b) shows the regret performance of DPBE and DP-DPBE with privacy parameters $\varepsilon = 15$ and $\delta = 10^{-6}$. We observe that although DP-DPBE adds extra noise to protect privacy, it can still achieve no-regret (i.e., $\lim_{T\to\infty} \frac{R(T)}{T} \to 0$). Indeed, to protect privacy, DP-DPBE requires much more time to find the optimal action, which is the typical regret-privacy tradeoff. However, for a large T, the gap compared to the non-private one is small, which also validates the privacy "for-free" result.

7.2 Standard Benchmark Functions

In addition, we study the performance of DPBE on standard optimization benchmark functions. This corresponds to a more realistic setting where the RKHS norm of the target function is unknown in advance. In particular, we use three common functions in global optimization problems [37]: (a) Sphere function, (b) Six-hump Camel function, and (c) Michalewicz function, and provide the performance comparison of DPBE-Fixed, DPBE, and DPBE-NoBatching in Figure 6 and Table 3. In the simulations, we scale the range of the function values to $[-1, 1]$ and use RKHS norm $B = 1$ in the algorithms as in [18]. Without knowing the exact kernel of the target function, each local reward function f_u is constructed by sampling a function from the GP $\mathcal{GP}(f(\cdot), v^2 k(\cdot, \cdot))$, where



(a) Sphere function (b) Six-Hump Camel function

(c) Michalewicz function (d) Light sensor data

Fig. 6. Comparison of regret performance under DPBE, DPBE-Fixed, and DPBE-NoBatching on four functions. (a) Sphere function. Settings: $d = 3, C = 1.6, \sigma = 0.01, \lambda = \sigma^2/v^2, \alpha = 0.7$; (b) Six-Hump Camel function. Settings: $d = 2, C = 1.6, \sigma = 0.01, \lambda = \sigma^2/v^2, \alpha = 0.7$; (c) Michalewicz function. Settings: $d = 2, C = 1.6, \sigma = 0.1, \lambda = \sigma^2/v^2, \alpha = 0.6$; (d) Function from light sensor data. Settings: $d = 2, C = 1.42, \sigma = 0.01, \lambda = \sigma^2/v^2, \alpha = 0.8$.

we choose $v^2 = 0.001$ and use the SE kernel with $l_{SE} = 0.2$. In addition, we set $T = 4 \times 10^4$ and $|\mathcal{D}| = 100$ and run each algorithm on each function for 20 times.

From Figure 6 and Table 3, we observe similar results to those of the synthetic function with the same kernel. First, compared to DPBE-Fixed that incurs the same communication cost, DPBE might perform slightly worse at the very beginning (e.g., Figure 6(a)) but eventually achieves a much smaller regret. Note that DPBE-Fixed may not be able to find the optimal action by the end of T (e.g., Figure 6(b)). This phenomenon strengthens our argument on the exponentially-increasing user-sampling mechanism in DPBE. While DPBE-NoBatching has slightly better regret performance than DPBE, it incurs much higher communication cost ($5 \sim 13\times$) and requires a much longer time ($6 \sim 23\times$, see running time column in Table 3), which demonstrates the key benefits of the batching strategy in improving communication efficiency and computation complexity.

In addition, we also evaluate the regret-privacy tradeoff under DP-DPBE. Due to space limitations, we present the numerical results in Appendix F (see Figures 8 and 9).

7.3 Functions from Real-World Data

We also evaluate the performance of DPBE on a function from a real-world dataset, where there is no explicit closed-form expression.

Light Sensor Data. We use the light sensor data collected from the CMU Intelligent Workplace in November 2005, which is available online [34]. It contains locations of 41 sensors, 601 training

Table 3. Communication cost and running time under DPBE, DPBE-Fixed, and DPBE-NoBatching

Function	Algorithm	Communication cost	Running time (seconds)
Sphere	DPBE	1.49×10^3	0.07
	DPBE-Fixed	1.49×10^3	0.12
	DPBE-NoBatching	6.16×10^3	0.69
Six-Hump Camel	DPBE	1.26×10^3	0.03
	DPBE-Fixed	1.26×10^3	0.12
	DPBE-NoBatching	1.45×10^4	0.17
Michalewicz	DPBE	2.06×10^3	0.06
	DPBE-Fixed	2.06×10^3	0.14
	DPBE-NoBatching	2.73×10^4	0.49
Light Sensor Data	DPBE	5.17×10^3	0.22
	DPBE-Fixed	5.17×10^3	0.28
	DPBE-NoBatching	2.73×10^4	5.20

samples, and 192 testing samples. Following [12, 36, 41], we compute the empirical covariance matrix of the training samples and use it as the kernel matrix in the algorithm. Here, for each location $\mathbf{x}$, we let $f(\mathbf{x})$ be the average of the normalized sample readings at $\mathbf{x}$ and set $B = \max_{\mathbf{x}} f(\mathbf{x})$ in the algorithm. For this function (from real data), we construct each local function f_u by sampling a function from a Gaussian process with mean f and the kernel constructed above, and set the noise in the local feedback as $\sigma = 0.01$ and the bias in each local feedback as $v = 0.1$. We run DPBE with input parameters $\alpha = 0.7$, $\beta = 1/(|\mathcal{D}|T)$, and $\lambda = \sigma^2/v^2$, and present the regret performance in Figure 6(d) and communication cost and running time in Table 3. The observations are qualitatively similar to those made in simulations on other functions: DPBE outperforms DPBE-Fixed in regret given the same communication cost and achieves a regret close to DPBE-NoBatching, which has much longer running time. Besides, we also run DP-DPBE on this real-world dataset and present the results in Appendix F (see Figures 8(d) and 9(d)), which validates the regret-privacy tradeoff.

8 COMPARISON WITH THE STATE-OF-THE-ARTS

8.1 Discussion

We now consider an alternative way of addressing kernelized bandits with distributed biased feedback. One may incorporate the local bias as another level of noise added to the noise in the rewards as a new noisy measurement of the global function f with a larger variance. In this case, the state-of-the-art algorithms for the traditional kernelized bandits [12, 36] may be adapted to our setting. However, they have some key limitations.

Consider two representative state-of-the-art algorithms: GP-UCB [12] and BPE [25]. GP-UCB is one of the most commonly used algorithms for standard kernelized bandits, It was proposed in [36] and improved in [12]. By resorting to the Gaussian process surrogate model (see Section 3.3), GP-UCB adaptively selects the action with the maximal *upper confidence bound* in each round based on historical observations up to the current round. BPE is a batch-based algorithm that eliminates suboptimal actions batch by batch, and within each batch, actions are chosen independently from reward observations. In the following, we compare our proposed DPBE algorithm with GP-UCB and BPE (adapted to our setting) and show their limitations in user-sampling, communication cost, and computation complexity.

First, both GP-UCB and BPE require to collect feedback from one user per step, which results in T users involved in the learning process. In practice, even though there is a large population, not all

users are willing to send their feedback. Hence, it may not be feasible to collect feedback from too many users. In our algorithm, instead of sampling more users to reduce the overall uncertainty, we ask each sampled user (who is more willing to participate) to participate in more rounds and send their feedback. In this way, we alleviate the user-sampling burden by letting the participating users collect more reward samples of the chosen actions. However, due to the bias in the feedback of each user, we could not just sample one user and then let her report the feedback during the entire horizon. We need to balance the tradeoff between sampling more users and letting the users participate in more rounds.

Second, by collecting feedback in each round, both GP-UCB and BPE incur a very high communication cost of T. Instead, we employ a phase-based communication protocol where feedback corresponding to any particular action at each participant is averaged and only communicated at the end of each phase. Then, the total communication cost depends on the number of phases, the number of distinct actions in each phase, and the number of sampled users. The smaller each of these three factors, the smaller the communication cost. By carefully designing the algorithm, we can reduce the communication cost to $O(\min\{\gamma_T, |\mathcal{D}|\}T^\alpha)$, where $\alpha \in (0, 1)$ is the user-sampling parameter one can choose.

Finally, at each round t, GP-UCB finds the decision action $\mathbf{x}_t$ that maximizes an acquisition function (specifically, the UCB index, which is the sum of the posterior mean and variance). Note that obtaining the posterior mean and variance requires computing matrix inverse (see Eqs. (4) and (5)), which still has a computation complexity of $O(t^2)$ even using rank-one recursive updates [12, Appendix 7]. Hence, the overall computation complexity of GP-UCB is $O(|\mathcal{D}|T^3)$. Similarly, BPE may also compute the posterior variance using the rank-one recursive update within each batch, and then the total computation complexity depends on the batch size and the number of batches. As in [25], the batch size is updated as $N_i = \sqrt{T\sqrt{N_{i-1}}}$, initialized with $N_0 = 1$, which results in $\lceil \log\log(T) \rceil$ batches in total. Therefore, the computation complexity of BPE is $O(|\mathcal{D}|T^3)$. In our design, we employ the batch schedule strategy and reformulate the posterior mean and variance as Eqs. (11) and (7), where the dimension of the matrix becomes much smaller. This leads to a much smaller overall computation complexity of $O(\gamma_T T^\alpha + (|\mathcal{D}|\gamma_T^3 + \gamma_T^4)\log T)$.

8.2 Empirical Performance

In this subsection, we evaluate the empirical performance of DPBE with different values of user-sampling parameter α compared to GP-UCB and BPE.

The simulations are run on the same three types of functions as in the preceding section: the synthetic function in Section 7.1, the standard benchmark functions in Section 7.2, and the function from light sensor data in Section 7.3. Due to space limitation, we only present the results of the synthetic function here and put the results of the latter two types of functions in Appendix F.

Consider[4] $\alpha \in \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ for DPBE. We show the empirical regret performance of all algorithms in Figure 7 and the running time in Table 4. From Figure 7, we observe that the empirical regret performance of DPBE can be fairly close to or even better than that of GP-UCB and BPE via properly choosing parameter α. However, it consumes much less time for DPBE with each $\alpha \in \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ than both GP-UCB and BPE. For example, while DPBE takes about 0.15 second in most scenarios, GP-UCB takes more than 5 seconds, which is more than 30 times slower. BPE takes around 27 seconds, which is even slower.

[4]Note that the smaller the value of α, the larger the cumulative regret. In Figure 7, we omit the regret performance when $\alpha < 0.4$ since they are much larger than others.

Table 4. Comparison of running time (seconds) under GP-UCB, BPE, and DPBE with different values of α.

Algorithms	DPBE						GP-UCB	BPE
	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 0.8$	$\alpha = 0.9$		
Running time	0.24	0.19	0.14	0.12	0.13	0.17	5.32	27.49

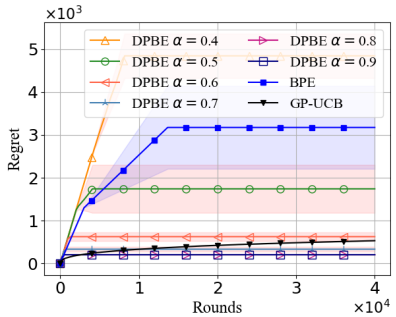


Fig. 7. Regret performance comparison of GP-UCB, BPE, and DPBE with different values of α.

Recall the empirical communication cost of DPBE with different values of α shown in Figure 4. While the communication cost of GP-UCB and BPE is 4×10^4 (specifically, one feedback per round), DPBE incurs a much smaller communication cost even when $\alpha = 0.9$ (4×10^4 vs. 1.19×10^4).

In summary, the comparison of empirical performance under DPBE with GP-UCB and BPE demonstrates the significant improvements of DPBE in terms of communication cost and computation complexity, although little regret performance is sacrificed when α is not big enough.

9 CONCLUSION

In this paper, we studied a new kernelized bandit problem with distributed biased feedback, where the feedback of the unknown objective function is biased due to user heterogeneity. To learn and optimize the unknown function using distributed biased feedback, we proposed the learning with communication framework. Considering the communication cost for collecting feedback and the computational bottleneck of kernelized bandits, we carefully designed the distributed phase-then-batch-based elimination (DPBE) algorithm to address all the new challenges. Specifically, DPBE selects actions according to maximum variance reduction, reduces bias via user-sampling, and improves communication efficiency and computation complexity via the batching strategy. Furthermore, we showed that DPBE achieves a sublinear regret while being scalable in terms of communication efficiency and computation complexity. Finally, we generalized DPBE to incorporate various differential privacy models to ensure privacy guarantees for participating users.

Future work. While we proposed a new DPBE algorithm to address the new challenges that arise in our problem setup, it would be worthwhile to explore other batch-based algorithms and investigate whether one can further improve the tradeoff among regret, communication efficiency, and computation complexity. In addition, as discussed in Remark 5.4, the lower bound derived for the standard kernelized bandits is also a valid lower bound for our problem. We show that our algorithm, if sampling a sufficient number of users, can achieve this lower bound. In general, however, it is an important open problem to close the gap by deriving tighter lower and/or upper bounds that capture the effect of user sampling in our new setting. We leave it as our future work.

ACKNOWLEDGMENTS

We thank our shepherd, Giulia Fanti, and the anonymous paper reviewers for their insightful feedback. We also thank Duo Cheng for fruitful discussions. This work is supported in part by the NSF grants under CNS-2112694 and CNS-2153220.

REFERENCES

[1] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. 2011. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems* 24 (2011).
[2] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. 2017. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th symposium on operating systems principles*. 441–459.
[3] Ilija Bogunovic and Andreas Krause. 2021. Misspecified Gaussian Process Bandit Optimization. *Advances in Neural Information Processing Systems* 34 (2021).
[4] Ilija Bogunovic, Zihan Li, Andreas Krause, and Jonathan Scarlett. 2022. A Robust Phased Elimination Algorithm for Corruption-Tolerant Gaussian Process Bandits. *arXiv preprint arXiv:2202.01850* (2022).
[5] Djallel Bouneffouf, Irina Rish, and Charu Aggarwal. 2020. Survey on applications of multi-armed and contextual bandits. In *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 1–8.
[6] Daniele Calandriello, Luigi Carratino, Alessandro Lazaric, Michal Valko, and Lorenzo Rosasco. 2020. Near-linear time Gaussian process optimization with adaptive batching and resparsification. In *International Conference on Machine Learning*. PMLR, 1295–1305.
[7] Daniele Calandriello, Luigi Carratino, Alessandro Lazaric, Michal Valko, and Lorenzo Rosasco. 2022. Scaling Gaussian Process Optimization by Evaluating a Few Unique Candidates Multiple Times. *arXiv preprint arXiv:2201.12909* (2022).
[8] Romain Camilleri, Kevin Jamieson, and Julian Katz-Samuels. 2021. High-dimensional experimental design and kernel bandits. In *International Conference on Machine Learning*. PMLR, 1227–1237.
[9] Mingzhe Chen, Nir Shlezinger, H Vincent Poor, Yonina C Eldar, and Shuguang Cui. 2021. Communication-efficient federated learning. *Proceedings of the National Academy of Sciences* 118, 17 (2021).
[10] Albert Cheu, Matthew Joseph, Jieming Mao, and Binghui Peng. 2021. Shuffle private stochastic convex optimization. *arXiv preprint arXiv:2106.09805* (2021).
[11] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. 2019. Distributed differential privacy via shuffling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 375–403.
[12] Sayak Ray Chowdhury and Aditya Gopalan. 2017. On kernelized multi-armed bandits. In *International Conference on Machine Learning*. PMLR, 844–853.
[13] Zhongxiang Dai, Bryan Kian Hsiang Low, and Patrick Jaillet. 2020. Federated Bayesian optimization via Thompson sampling. *Advances in Neural Information Processing Systems* 33 (2020), 9687–9699.
[14] Yihan Du, Wei Chen, Yuko Yuroki, and Longbo Huang. 2021. Collaborative Pure Exploration in Kernel Bandit. *arXiv preprint arXiv:2110.15771* (2021).
[15] Abhimanyu Dubey. 2021. No-regret algorithms for private gaussian process bandit optimization. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2062–2070.
[16] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.
[17] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.* 9, 3-4 (2014), 211–407.
[18] David Janz, David Burt, and Javier González. 2020. Bandit optimisation of functions in the Matérn kernel RKHS. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2486–2495.
[19] Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K Sriperumbudur. 2018. Gaussian processes and kernel methods: A review on connections and equivalences. *arXiv preprint arXiv:1807.02582* (2018).
[20] Tor Lattimore and Csaba Szepesvári. 2020. *Bandit algorithms*. Cambridge University Press.
[21] Tor Lattimore, Csaba Szepesvari, and Gellert Weisz. 2020. Learning with good feature representations in bandits and in rl with a generative model. In *International Conference on Machine Learning*. PMLR, 5662–5670.
[22] Fengjiao Li, Xingyu Zhou, and Bo Ji. 2022. Differentially Private Linear Bandits with Partial Distributed Feedback. *arXiv preprint arXiv:2207.05827* (2022).
[23] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*. 661–670.
[24] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2017. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research* 18, 1 (2017),

6765–6816.
[25] Zihan Li and Jonathan Scarlett. 2022. Gaussian process bandit optimization with few batches. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 92–107.
[26] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. 2020. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials* 22, 3 (2020), 2031–2063.
[27] Ajay Mahimkar, Ashiwan Sivakumar, Zihui Ge, Shomik Pathak, and Karunasish Biswas. 2021. Auric: using data-driven recommendation to automatically generate cellular configuration. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. 807–820.
[28] Nikita Mishra and Abhradeep Thakurta. 2015. (Nearly) optimal differentially private stochastic multi-arm bandits. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*. 592–601.
[29] Kanishka Misra, Eric M Schwartz, and Jacob Abernethy. 2019. Dynamic online pricing with incomplete information using multiarmed bandit experiments. *Marketing Science* 38, 2 (2019), 226–252.
[30] Carl Edward Rasmussen. 2003. Gaussian processes in machine learning. In *Summer school on machine learning*. Springer, 63–71.
[31] Carl Edward Rasmussen and Christopher KI Williams. 2006. *Gaussian processes for machine learning*. Vol. 1. MIT press Cambridge, MA.
[32] Sayak Ray Chowdhury and Aditya Gopalan. 2019. Bayesian optimization under heavy-tailed payoffs. *Advances in Neural Information Processing Systems* 32 (2019).
[33] Touqir Sajed and Or Sheffet. 2019. An optimal private stochastic-mab algorithm based on optimal private stopping rule. In *International Conference on Machine Learning*. PMLR, 5579–5588.
[34] School of Computer Science, Carnegie Mellon University [n.d.]. Light sensor data. Retrieved October 05, 2022, from http://www.cs.cmu.edu/~guestrin/Class/10708-F08/projects/lightsensor.zip.
[35] Rachael Hwee Ling Sim, Yehong Zhang, Bryan Kian Hsiang Low, and Patrick Jaillet. 2021. Collaborative Bayesian optimization with fair regret. In *International Conference on Machine Learning*. PMLR, 9691–9701.
[36] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. 2009. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995* (2009).
[37] S. Surjanovic and D. Bingham. [n.d.]. Virtual Library of Simulation Experiments: Test Functions and Datasets. Retrieved July 29, 2022, from http://www.sfu.ca/~ssurjano.
[38] Jay Tenenbaum, Haim Kaplan, Yishay Mansour, and Uri Stemmer. 2021. Differentially private multi-armed bandits in the shuffle model. *Advances in Neural Information Processing Systems* 34 (2021), 24956–24967.
[39] Sattar Vakili, Nacime Bouziani, Sepehr Jalali, Alberto Bernacchia, and Da-shan Shiu. 2021. Optimal order simple regret for gaussian process bandits. *Advances in Neural Information Processing Systems* 34 (2021).
[40] Sattar Vakili, Kia Khezeli, and Victor Picheny. 2021. On information gain and regret bounds in gaussian process bandits. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 82–90.
[41] Xingyu Zhou and Bo Ji. 2022. On Kernelized Multi-Armed Bandits with Constraints. *arXiv preprint arXiv:2203.15589* (2022).
[42] Xingyu Zhou and Jian Tan. 2020. Local differential privacy for bayesian optimization. *arXiv preprint arXiv:2010.06709* (2020).
[43] Yinglun Zhu, Dongruo Zhou, Ruoxi Jiang, Quanquan Gu, Rebecca Willett, and Robert Nowak. 2021. Pure exploration in kernel and neural bandits. *Advances in Neural Information Processing Systems* 34 (2021), 11618–11630.

Table 5. Bounds on γ_T and Regret under Two Common Kernels [40]

Kernel	Upper Bound on γ_T	Regret Lower Bound	Regret Upper Bound $O(\sqrt{\gamma_T T})$
SE	$O\left(\log^{d+1}(T)\right)$	$\Omega\left(\sqrt{T\log^{\frac{d}{2}}(T)}\right)$	$O\left(\sqrt{T\log^{d+1}(T)}\right)$
Matérn-ν	$O\left(T^{\frac{d}{2\nu+d}}\log^{\frac{2\nu}{2\nu+d}}(T)\right)$	$\Omega\left(T^{\frac{\nu+d}{2\nu+d}}\right)$	$O\left(T^{\frac{\nu+d}{2\nu+d}}\log^{\frac{\nu}{2\nu+d}}(T)\right)$

A KERNELIZED BANDITS: USEFUL DEFINITIONS AND USEFUL RESULTS

A.1 Example Kernel Functions

In the following, we list some commonly used kernel functions $k : \mathcal{D}\times\mathcal{D}\to\mathbb{R}$:

- Linear kernel: $k_{\text{lin}}(\mathbf{x},\mathbf{x}') = \mathbf{x}^\top\mathbf{x}'$,
- Squared exponential kernel: $k_{\text{SE}}(\mathbf{x},\mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|}{2l^2}\right)$,
- Matérn kernel: $k_{\text{Mat}}(\mathbf{x},\mathbf{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}\|\mathbf{x}-\mathbf{x}'\|}{l}\right)J_\nu\left(\frac{\sqrt{2\nu}\|\mathbf{x}-\mathbf{x}'\|}{l}\right)$,

where l denotes the length-scale hyperparameter, $\nu > 0$ is an additional hyperparameter that dictates the smoothness, and J_ν and Γ_ν denote the modified Bessel function and the Gamma function, respectively [31].

A.2 Maximum Information Gain for Different Kernels

We present the bounds on γ_T and regret under two common kernels below in Table 5.

A.3 Useful Results

Lemma A.1 (Sum of variance. Lemma 6 in [32]). *Let $\mathbf{X}_t = [\mathbf{x}_1^\top,\ldots,\mathbf{x}_t^\top]^\top$, and $\sigma_t^2(\mathbf{x}) \triangleq k(\mathbf{x},\mathbf{x}) - \mathbf{k}(\mathbf{x},\mathbf{X}_t)^\top(\mathbf{K}_{\mathbf{X}_t\mathbf{X}_t}+\lambda\mathbf{I})^{-1}\mathbf{k}(\mathbf{x},\mathbf{X}_t)$ for any $\mathbf{x}\in\mathcal{D}$. Then, we have*

$$\sum_{s=1}^{t}\sigma_s^2(x_s) \leqslant \lambda\ln|\lambda^{-1}\mathbf{K}_{\mathbf{X}_t\mathbf{X}_t}+\mathbf{I}| \leqslant 2\lambda\gamma_t. \tag{19}$$

Lemma A.2 (Proposition A.1 in [7]/Lemma 4 in [6]). *For any kernel k, set of points $\mathbf{X}_\tau$, $\mathbf{x}\in\mathcal{D}$, and $\tau' < \tau$, we have*

$$1 \leqslant \frac{\sigma_{\tau'}^2(\mathbf{x})}{\sigma_\tau^2(\mathbf{x})} \leqslant 1+\sum_{s=\tau'+1}^{\tau}\sigma_{\tau'}^2(\mathbf{x}_s). \tag{20}$$

A.4 Formulation in Feature Space

For several of the proofs, it will be useful to introduce the so-called feature space (RKHS) formulation of any point in the primal space $\mathbb{R}^d$. In particular, we define a feature map $\varphi(\mathbf{x}) = k(\mathbf{x},\cdot)$ where $\varphi : \mathcal{D}\to\mathcal{H}_k$ with $\mathcal{H}_k$ being the reproducing kernel Hilbert space (RKHS) associated with kernel function k. According to the properties of RKHS, we have the following observations:

- For any $\mathbf{x},\mathbf{x}'$, $k(\mathbf{x},\mathbf{x}') = \varphi(\mathbf{x})^\top\varphi(\mathbf{x}')$.
- For any function $f\in\mathcal{H}$, $f(\mathbf{x}) = \langle f,\varphi(\mathbf{x})\rangle = \varphi(\mathbf{x})^\top f$.
- Fundamental linear algebra equality

$$(BB^\top+\lambda\mathbf{I})^{-1}B = B(B^\top B+\lambda\mathbf{I})^{-1}. \tag{21}$$

- Define $\Phi_h \triangleq [\varphi(\mathbf{a}_1)^\top, \ldots, \varphi(\mathbf{a}_h)^\top]^\top$. Then, the kernel matrix $\mathbf{K}_{\mathbf{A}_h\mathbf{A}_h} = \Phi_h\Phi_h^\top$ and $k(\mathbf{x}, \mathbf{A}_h) = \Phi_h\varphi(\mathbf{x})$, and the variance function $\Sigma_h^2(\cdot)$ represented in the feature space is the following:

$$\begin{aligned}\Sigma_h^2(\mathbf{x}) &= k(\mathbf{x},\mathbf{x}) - \mathbf{k}(\mathbf{x},\mathbf{A}_h)^\top(\mathbf{K}_{\mathbf{A}_h\mathbf{A}_h} + \lambda\mathbf{W}_h^{-1})^{-1}\mathbf{k}(\mathbf{x},\mathbf{A}_h)\\ &= \varphi(\mathbf{x})^\top\varphi(\mathbf{x}) - \varphi(\mathbf{x})^\top\Phi_h^\top(\Phi_h\Phi_h^\top + \lambda\mathbf{W}_h^{-1})^{-1}\Phi_h\varphi(\mathbf{x}).\end{aligned} \tag{22}$$

- Consider any phase l. Recall that H_l is the number of batches in the l-th phase. Define $\Phi_{H_l} \triangleq [\varphi(\mathbf{a}_1)^\top, \ldots, \varphi(\mathbf{a}_{H_l})^\top]^\top$. Then, the kernel matrix $\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} = \Phi_{H_l}\Phi_{H_l}^\top$ and $k(\mathbf{x}, \mathbf{A}_{H_l}) = \Phi_{H_l}\varphi(\mathbf{x})$.
- Define $\Phi_\tau \triangleq [\varphi(\mathbf{x}_{t_l+1})^\top, \ldots, \varphi(\mathbf{x}_{t_l+\tau})^\top]^\top$. Then, the kernel matrix $\mathbf{K}_{\mathbf{X}_\tau\mathbf{X}_\tau} = \Phi_\tau\Phi_\tau^\top$, $k(\mathbf{x}, \mathbf{X}_\tau) = \Phi_\tau\varphi(\mathbf{x})$, and the variance function $\sigma_\tau^2(\cdot)$ represented in the feature space is the following:

$$\begin{aligned}\sigma_\tau^2(\mathbf{x}) &= k(\mathbf{x},\mathbf{x}) - \mathbf{k}(\mathbf{x},\mathbf{X}_\tau)^\top(\mathbf{K}_{\mathbf{X}_\tau\mathbf{X}_\tau} + \lambda\mathbf{I})^{-1}\mathbf{k}(\mathbf{x},\mathbf{X}_\tau)\\ &= \varphi(\mathbf{x})^\top\varphi(\mathbf{x}) - \varphi(\mathbf{x})^\top\Phi_\tau^\top(\Phi_\tau\Phi_\tau^\top + \lambda\mathbf{I})^{-1}\Phi_\tau\varphi(\mathbf{x})\\ &= \varphi(\mathbf{x})^\top\varphi(\mathbf{x}) - \varphi(\mathbf{x})^\top(\Phi_\tau^\top\Phi_\tau + \lambda\mathbf{I})^{-1}\Phi_\tau^\top\Phi_\tau\varphi(\mathbf{x})\\ &= \varphi(\mathbf{x})^\top(\Phi_\tau^\top\Phi_\tau + \lambda\mathbf{I})^{-1}(\Phi_\tau^\top\Phi_\tau + \lambda\mathbf{I})\varphi(\mathbf{x}) - \varphi(\mathbf{x})^\top(\Phi_\tau^\top\Phi_\tau + \lambda\mathbf{I})^{-1}\Phi_\tau^\top\Phi_\tau\varphi(\mathbf{x})\\ &= \lambda\varphi(\mathbf{x})^\top(\Phi_\tau^\top\Phi_\tau + \lambda\mathbf{I})^{-1}\varphi(\mathbf{x}).\end{aligned} \tag{23}$$

- Define $\Phi_{T_l} \triangleq [\varphi(\mathbf{x}_{t_l+1})^\top, \ldots, \varphi(\mathbf{x}_{t_l+T_l})^\top]^\top$. Then, the kernel matrix $\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} = \Phi_{T_l}\Phi_{T_l}^\top$, $k(\mathbf{x}, \mathbf{X}_{T_l}) = \Phi_{T_l}\varphi(\mathbf{x})$, and $f(\mathbf{X}_{T_l}) = \Phi_{T_l}f$.

B AUXILIARY RESULTS AND PROOFS FOR REGRET ANALYSIS

B.1 Equivalent Representations

Consider any phase l. We use τ to denote the within-phase time index, i.e., $\tau \in \{1, \cdots, T_l\}$. Define τ_h as the last within-phase time index of the h-th batch, i.e., $\tau_h \triangleq \max\{\tau : t_l + \tau \in \mathcal{T}_l(\mathbf{a}_h)\}$. Then, after playing τ_h actions, the posterior variance in the traditional GP model is the following:

$$\sigma_{\tau_h}^2(\mathbf{x}) = k(\mathbf{x},\mathbf{x}) - \mathbf{k}(\mathbf{x},\mathbf{X}_{\tau_h})^\top(\mathbf{K}_{\mathbf{X}_{\tau_h}\mathbf{X}_{\tau_h}} + \lambda\mathbf{I})^{-1}\mathbf{k}(\mathbf{x},\mathbf{X}_{\tau_h}). \tag{24}$$

For the posterior mean, without the observations $\mathbf{y}_{T_l} = [y_{t_l+1}, \ldots, y_{t_l+T_l}]^\top$ corresponding to the actions $\mathbf{X}_{T_l} = [\mathbf{x}_{t_l+1}^\top \ldots, \mathbf{x}_{t_l+T_l}^\top]^\top$, we replace $\mathbf{y}_{T_l}$ with $\frac{1}{|U_l|}\sum_{u\in U_l}\mathbf{y}_{l,u}$ where $\mathbf{y}_{l,u} = [y_{u,t_l+1}, \ldots, y_{u,t_l+T_l}]^\top$ in the traditional GP model. Then, the posterior mean becomes the following:

$$\mu_{T_l}(\mathbf{x}) = \frac{1}{|U_l|}\sum_{u\in U_l}\mathbf{k}(\mathbf{x},\mathbf{X}_{T_l})^\top(\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda\mathbf{I})^{-1}\mathbf{y}_{l,u}. \tag{25}$$

In our algorithm, in order to save computation complexity and communication cost, we use Eq. (7) and Eq. (11) instead of the above formula. In the following lemma, we show that they are equivalent.

Lemma B.1 (Equivalent representations). *Consider any phase l. By the end of the h-th phase, the posterior variance Eq. (24) in the traditional GP model is equivalent to Eq. (7) used in our* DPBE *algorithm. That is, for any $\mathbf{x} \in \mathcal{D}$, we have*

$$\sigma_{\tau_h}^2(\mathbf{x}) = \Sigma_h^2(\mathbf{x}), \quad \forall h = 1, \ldots, H_l. \tag{26}$$

Moreover, we have the two representations (Eq. (25) and Eq. (11)) for the posterior mean function are equivalent. That is, for any $\mathbf{x} \in \mathcal{D}$, we have

$$\mu_{T_l}(\mathbf{x}) = \bar{\mu}_l(\mathbf{x}). \tag{27}$$

Proof. First, we have the following result, which helps connect the two representations of mean and variance functions:

$$\begin{aligned}\Phi_{\tau_h}^\top \Phi_{\tau_h} &= \sum_{t=t_l+1}^{t_l+\tau_h} \varphi(\mathbf{x}_t)\varphi(\mathbf{x}_t)^\top \\ &\overset{(a)}{=} \sum_{i=1}^{h} T_l(\mathbf{a}_i)\varphi(\mathbf{a}_i)\varphi(\mathbf{a}_i)^\top \\ &= \Phi_h^\top \mathbf{W}_h \Phi_h,\end{aligned} \tag{28}$$

where (a) is due to our algorithm decisions: $\mathbf{x}_t = \mathbf{a}_i$ for any $t \in \mathcal{T}_l(\mathbf{a}_i) = \{t_l+\tau_{i-1}+1, t_l+\tau_{i-1}+T_l(\mathbf{a}_i)\}$ and the last step holds because $\mathbf{W}_h$ is a diagonal matrix with $(\mathbf{W}_h)_{ii} = T_l(\mathbf{a}_i)$ for any $i \in [h]$.

Then, we are ready to derive the equivalence of two representations of the mean function.

1) Variance representation equivalence: $\sigma^2_{\tau_h}(\mathbf{x}) = \Sigma^2_h(\mathbf{x})$ for $h = 1, \ldots, H_l$. This implies

$$\begin{aligned}&k(\mathbf{x},\mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{X}_{\tau_h})^\top (\mathbf{K}_{\mathbf{X}_{\tau_h}\mathbf{X}_{\tau_h}} + \lambda \mathbf{I})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{X}_{\tau_h}) \\ &= k(\mathbf{x},\mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{A}_h)^\top (\mathbf{K}_{\mathbf{A}_h\mathbf{A}_h} + \lambda \mathbf{W}_h^{-1})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{A}_h).\end{aligned}$$

It remains to show the following:

$$\mathbf{k}(\mathbf{x}, \mathbf{X}_{\tau_h})^\top (\mathbf{K}_{\mathbf{X}_{\tau_h}\mathbf{X}_{\tau_h}} + \lambda \mathbf{I})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{X}_{\tau_h}) = \mathbf{k}(\mathbf{x}, \mathbf{A}_h)^\top (\mathbf{K}_{\mathbf{A}_h\mathbf{A}_h} + \lambda \mathbf{W}_h^{-1})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{A}_h). \tag{29}$$

Using the feature space formulations, we have

$$\begin{aligned}&\mathbf{k}(\mathbf{x}, \mathbf{A}_h)^\top (\mathbf{K}_{\mathbf{A}_h\mathbf{A}_h} + \lambda \mathbf{W}_h^{-1})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{A}_h) \\ =&\varphi(\mathbf{x})^\top \Phi_h^\top (\Phi_h \Phi_h^\top + \lambda \mathbf{W}_h^{-1})^{-1} \Phi_h \varphi(\mathbf{x}) \\ =&\varphi(\mathbf{x})^\top \Phi_h^\top \mathbf{W}_h^{1/2} (\mathbf{W}_h^{1/2} \Phi_h \Phi_h^\top \mathbf{W}_h^{1/2} + \lambda \mathbf{I})^{-1} \mathbf{W}_h^{1/2} \Phi_h \varphi(\mathbf{x}) \\ =&\varphi(\mathbf{x})^\top (\Phi_h^\top \mathbf{W}_h^{1/2} \mathbf{W}_h^{1/2} \Phi_h + \lambda \mathbf{I})^{-1} \Phi_h^\top \mathbf{W}_h^{1/2} \mathbf{W}_h^{1/2} \Phi_h \varphi(\mathbf{x}) \\ =&\varphi(\mathbf{x})^\top (\Phi_h^\top \mathbf{W}_h \Phi_h + \lambda \mathbf{I})^{-1} \Phi_h^\top \mathbf{W}_h \Phi_h \varphi(\mathbf{x}) \\ \overset{(a)}{=}&\varphi(\mathbf{x})^\top (\Phi_{\tau_h}^\top \Phi_{\tau_h} + \lambda \mathbf{I})^{-1} \Phi_{\tau_h}^\top \Phi_{\tau_h} \varphi(\mathbf{x}) \\ =&\varphi(\mathbf{x})^\top \Phi_{\tau_h}^\top (\Phi_{\tau_h} \Phi_{\tau_h}^\top + \lambda \mathbf{I})^{-1} \Phi_{\tau_h} \varphi(\mathbf{x}) \\ =&\mathbf{k}(\mathbf{x}, \mathbf{X}_{\tau_h})^\top (\mathbf{K}_{\mathbf{X}_{\tau_h}\mathbf{X}_{\tau_h}} + \lambda \mathbf{I})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{X}_{\tau_h}),\end{aligned} \tag{30}$$

where (a) is from Eq. (28). Then, we have $\sigma^2_{\tau_h}(\mathbf{x}) = \Sigma^2_h(\mathbf{x})$.

2) Mean representation equivalence: $\mu_{T_l}(\mathbf{x}) = \bar{\mu}_l(\mathbf{x})$, i.e.,

$$\frac{1}{|U_l|} \sum_{u \in U_l} \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l})^\top (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda \mathbf{I})^{-1} \mathbf{y}_{l,u} = k(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \bar{\mathbf{y}}_l. \tag{31}$$

For the last within-phase index $\tau_{H_l} = T_l$, we also have the following:

$$\begin{aligned}
\frac{1}{|U_l|}\sum_{u\in U_l}\Phi_{T_l}^\top \mathbf{y}_{l,u} &= \frac{1}{|U_l|}\sum_{u\in U_l}\sum_{t=t_l+1}^{t_l+T_l} y_{u,t}\varphi(\mathbf{x}_t)\\
&= \frac{1}{|U_l|}\sum_{u\in U_l}\sum_{h=1}^{H_l}\sum_{t\in\mathcal{T}_l(\mathbf{a}_h)} y_{u,t}\varphi(\mathbf{x}_t)\\
&= \frac{1}{|U_l|}\sum_{u\in U_l}\sum_{h=1}^{H_l}\varphi(\mathbf{a}_h)\sum_{t\in\mathcal{T}_l(\mathbf{a}_h)} y_{u,t}\\
&= \frac{1}{|U_l|}\sum_{u\in U_l}\sum_{h=1}^{H_l}\varphi(\mathbf{a}_h)T_l(\mathbf{a}_h)y_l^u(\mathbf{a}_h)\\
&= \sum_{h=1}^{H_l} T_l(\mathbf{a}_h)y_l(\mathbf{a}_h)\varphi(\mathbf{a}_h)\\
&= \Phi_{H_l}^\top \mathbf{W}_{H_l}\bar{\mathbf{y}}_l.
\end{aligned} \tag{32}$$

Then, we are ready to derive the equivalence of two representations of the mean function:

$$\begin{aligned}
&\frac{1}{|U_l|}\sum_{u\in U_l}\mathbf{k}(\mathbf{x},\mathbf{X}_{T_l})^\top(\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}}+\lambda\mathbf{I})^{-1}\mathbf{y}_{l,u}\\
&= \frac{1}{|U_l|}\sum_{u\in U_l}\varphi(\mathbf{x})^\top\Phi_{T_l}^\top(\Phi_{T_l}\Phi_{T_l}^\top+\lambda\mathbf{I})^{-1}\mathbf{y}_{l,u}\\
&= \frac{1}{|U_l|}\sum_{u\in U_l}\varphi(\mathbf{x})^\top(\Phi_{T_l}^\top\Phi_{T_l}+\lambda\mathbf{I})^{-1}\Phi_{T_l}^\top\mathbf{y}_{l,u}\\
&= \varphi(\mathbf{x})^\top(\Phi_{T_l}^\top\Phi_{T_l}+\lambda\mathbf{I})^{-1}\cdot\frac{1}{|U_l|}\sum_{u\in U_l}\Phi_{T_l}^\top\mathbf{y}_{l,u}\\
&\overset{(a)}{=} \varphi(\mathbf{x})^\top(\Phi_{H_l}^\top\mathbf{W}_{H_l}\Phi_{H_l}+\lambda\mathbf{I})^{-1}\Phi_{H_l}^\top\mathbf{W}_{H_l}\bar{\mathbf{y}}_l\\
&= \varphi(\mathbf{x})^\top(\Phi_{H_l}^\top\mathbf{W}_{H_l}^{1/2}\mathbf{W}_{H_l}^{1/2}\Phi_{H_l}+\lambda\mathbf{I})^{-1}\Phi_{H_l}^\top\mathbf{W}_{H_l}^{1/2}\mathbf{W}_{H_l}^{1/2}\bar{\mathbf{y}}_l\\
&= \varphi(\mathbf{x})^\top((\mathbf{W}_{H_l}^{1/2}\Phi_{H_l})^\top(\mathbf{W}_{H_l}^{1/2}\Phi_{H_l})+\lambda\mathbf{I})^{-1}(\mathbf{W}_{H_l}^{1/2}\Phi_{H_l})^\top\mathbf{W}_{H_l}^{1/2}\bar{\mathbf{y}}_l\\
&= \varphi(\mathbf{x})^\top\Phi_{H_l}^\top\mathbf{W}_{H_l}^{1/2}(\mathbf{W}_{H_l}^{1/2}\Phi_{H_l}\Phi_{H_l}^\top\mathbf{W}_{H_l}^{1/2}+\lambda\mathbf{I})^{-1}\mathbf{W}_{H_l}^{1/2}\bar{\mathbf{y}}_l\\
&= \varphi(\mathbf{x})^\top\Phi_{H_l}^\top(\Phi_{H_l}\Phi_{H_l}^\top+\lambda\mathbf{W}_{H_l}^{-1})^{-1}\bar{\mathbf{y}}_l\\
&= \mathbf{k}(\mathbf{x},\mathbf{A}_{H_l})^\top(\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}}+\lambda\mathbf{W}_{H_l}^{-1})^{-1}\bar{\mathbf{y}}_l = \bar{\mu}_l(\mathbf{x}),
\end{aligned} \tag{33}$$

where (a) is from Eq. (28) with $\tau_{H_l} = T_l$ and the result in Eq. (32). □

B.2 Impact of Batch Schedule Strategy on Posterior Variance

In our batch schedule strategy, the decision $\mathbf{x}_t$ does not change for $T_l(\mathbf{a}_h)$ rounds when starting choosing $\mathbf{a}_h$ after τ_{h-1} rounds within the l-th phase. Applying Lemma A.2 to our setting with $\tau' = \tau_{h-1}$, we obtain the following corollary.

Corollary B.2. *Consider any phase l. Recall that τ_{h-1} is the within-phase time index before starting choosing $\mathbf{a}_h$. Then, give any set of chosen actions $\mathbf{A}_{h-1}$ for the first $h-1$ batches, for any kernel k, any*

$\mathbf{x} \in \mathcal{D}$, and any $\tau \in [\tau_{h-1}+1, \tau_{h-1}+T_l(\mathbf{a}_h)]$, we have

$$1 \leqslant \frac{\Sigma_{h-1}(\mathbf{x})}{\sigma_\tau(\mathbf{x})} \leqslant C. \tag{34}$$

Proof. Applying Lemma A.2 to our setting, we have

$$1 \leqslant \frac{\sigma^2_{\tau_{h-1}}(\mathbf{x})}{\sigma^2_\tau(\mathbf{x})} \leqslant 1 + T_l(\mathbf{a}_h)\sigma^2_{\tau_h}(\mathbf{a}_h). \tag{35}$$

Moreover, by selecting $T_l(\mathbf{a}_h) = \lfloor (C^2-1)/\Sigma^2_{h-1}(\mathbf{a}_h) \rfloor = \lfloor (C^2-1)/\sigma^2_{\tau_{h-1}}(\mathbf{a}_h) \rfloor$ (Lemma B.1) in our algorithm, we derive the result in Eq. (34). □

One key step to getting the regret upper bound is to bound the confidence width, which is related to the maximal value of the posterior variance by the end of each phase. (See Eq. (12). In the following, we provide a bound for the maximal value of the posterior variance.

Lemma B.3. *The posterior variance after H_l batches (decisions) in the l-th phase satisfies*

$$\max_{\mathbf{x}\in\mathcal{D}_l} \Sigma_{H_l}(\mathbf{x}) \leqslant \sqrt{\frac{2\sigma^2 C^2 \gamma_{T_l}}{T_l}}. \tag{36}$$

Proof. Recall that DPBE plays action $\mathbf{a}_h$ when $\tau \in [\tau_{h-1}+1, \tau_{h-1}+T_l(\mathbf{a}_h)]$ within the l-th phase. First, we have for any $\mathbf{x} \in \mathcal{D}_l$, any $h \leqslant H_l$,

$$\Sigma_{H_l}(\mathbf{x}) \overset{(a)}{\leqslant} \Sigma_{h-1}(\mathbf{x}) \overset{(b)}{\leqslant} \Sigma_{h-1}(\mathbf{a}_h) = \sigma_{\tau_{h-1}}(\mathbf{a}_h), \tag{37}$$

where (a) holds because $\Sigma_h(\cdot)$ is non-increasing in h, (b) is based on our decision, and the last step is due to the equivalent representation result. Then, we have the following:

$$\begin{aligned}
\max_{\mathbf{x}\in\mathcal{D}_l} \Sigma_{H_l}(\mathbf{x}) &\leqslant \frac{1}{T_l}\sum_{h=1}^{H_l} T_l(\mathbf{a}_h)\Sigma_{h-1}(\mathbf{a}_h) \\
&= \frac{1}{T_l}\sum_{h=1}^{H_l}\sum_{\tau=\tau_{h-1}+1}^{\tau_{h-1}+T_l(\mathbf{a}_h)} \Sigma_{h-1}(\mathbf{a}_h) \\
&= \frac{1}{T_l}\sum_{h=1}^{H_l}\sum_{\tau=\tau_{h-1}+1}^{\tau_{h-1}+T_l(\mathbf{a}_h)} \frac{\Sigma_{h-1}(\mathbf{a}_h)}{\sigma_\tau(\mathbf{a}_h)}\cdot\sigma_\tau(\mathbf{a}_h) \\
&\overset{(a)}{\leqslant} \frac{1}{T_l}\sum_{h=1}^{H_l}\sum_{\tau=\tau_{h-1}+1}^{\tau_{h-1}+T_l(\mathbf{a}_h)} C\sigma_\tau(\mathbf{a}_h) \\
&\overset{(b)}{=} \frac{C}{T_l}\sum_{h=1}^{H_l}\sum_{\tau=\tau_{h-1}+1}^{\tau_{h-1}+T_l(\mathbf{a}_h)} \sigma_\tau(\mathbf{x}_{t_l+\tau}) \\
&= \frac{C}{T_l}\sum_{\tau=1}^{T_l}\sigma_\tau(\mathbf{x}_{t_l+\tau}) \\
&\overset{(c)}{\leqslant} \frac{C}{T_l}\sqrt{T_l\sum_{\tau=1}^{T_l}\sigma^2_\tau(\mathbf{x}_{t_l+\tau})} \\
&\overset{(d)}{\leqslant} \frac{C}{T_l}\sqrt{T_l\cdot 2\lambda\gamma_{T_l}} = \sqrt{\frac{2\lambda C^2\gamma_{T_l}}{T_l}},
\end{aligned} \tag{38}$$

where the inequality (a) is from Corollary B.2, (b) is based on our algorithm decision: $\mathbf{x}_{t_l+\tau} = \mathbf{a}_h$ for any $\tau \in [\tau_{h-1}+1, \tau_{h-1}+T_l(\mathbf{a}_h)]$, (c) is by Cauchy-Schwartz inequality, and (d) is from Lemma A.1. □

B.3 Other Useful Results

Lemma B.4. *Consider any particular phase l. In the traditional GP models, without noise in the reward observations, the difference between the ground truth and regression estimator satisfies*

$$\left|f(\mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l})^\top (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda \mathbf{I})^{-1} f(\mathbf{X}_{T_l})\right| \leqslant B\sigma_{T_l}(\mathbf{x}). \tag{39}$$

Proof. Representing $f(\mathbf{x})$ in the feature space, we have

$$\begin{aligned}
&\left|f(\mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l})^\top (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda \mathbf{I})^{-1} f(\mathbf{X}_{T_l})\right| \\
&= \left|\varphi(\mathbf{x})^\top f - \varphi(\mathbf{x})^\top \Phi_{T_l}^\top (\Phi_{T_l}\Phi_{T_l}^\top + \lambda \mathbf{I})^{-1} \Phi_{T_l} f\right| \\
&= \left|\varphi(\mathbf{x})^\top f - \varphi(\mathbf{x})^\top (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I})^{-1} \Phi_{T_l}^\top \Phi_{T_l} f\right| \\
&= \left|\lambda \varphi(\mathbf{x})^\top (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I})^{-1} f\right| \\
&\leqslant \|f\|_k \|\lambda (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x})\|_k \\
&\leqslant B\sqrt{\lambda \varphi(\mathbf{x})^\top (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I})^{-1} \lambda \mathbf{I} (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x})} \\
&\leqslant B\sqrt{\lambda \varphi(\mathbf{x})^\top (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I})^{-1} (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I}) (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x})} \\
&\leqslant B\sqrt{\lambda \varphi(\mathbf{x})^\top (\Phi_{T_l}^\top \Phi_{T_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x})} \\
&= B\sigma_{T_l}(\mathbf{x}),
\end{aligned} \tag{40}$$

where the last step is from Eq. (23). □

C PROOFS OF THEOREM 5.1

Before proving Theorem 5.1, we first provide the key concentration inequality under DPBE in Theorem C.1.

Theorem C.1. *For any particular phase l, with probability at least $1 - 4\beta$, the following holds*

$$|f(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})| \leqslant w_l(\mathbf{x}), \tag{41}$$

where mean function $\bar{\mu}_l(\mathbf{x})$ and confidence width function $w_l(\mathbf{x})$ are defined in Eq. (11) and Eq. (12).

Proof. In this proof, we will show the following concentration inequality holds for any $\mathbf{x} \in \mathcal{D}$

$$\mathbb{P}[|f(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})| \geqslant w_l(\mathbf{x})] \leqslant 4\beta. \tag{42}$$

For any $\mathbf{x} \in \mathcal{D}$, we let $w_l(\mathbf{x}) = w_{l,1}(\mathbf{x}) + w_{l,2}(\mathbf{x})$, where

$$w_{l,1}(\mathbf{x}) \triangleq \sqrt{\frac{2k(\mathbf{x},\mathbf{x})\log(1/\beta)}{|U_l|}} \quad \text{and } w_{l,2}(\mathbf{x}) \triangleq \Sigma_{H_l}(\mathbf{x})\left(\sqrt{\frac{2\log(1/\beta)}{|U_l|}} + B\right).$$

First, for any $\mathbf{x} \in \mathcal{D}$, we have the following inequality:

$$|f(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})| \leqslant \left|f(\mathbf{x}) - \frac{1}{|U_l|}\sum_{u \in U_l} f_u(\mathbf{x})\right| + \left|\frac{1}{|U_l|}\sum_{u \in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})\right|.$$

Then, we have

$$\begin{aligned}
&\mathbb{P}\left[|f(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})| \geqslant w_l(\mathbf{x})\right] \\
\leqslant &\mathbb{P}\left[\left|f(\mathbf{x}) - \frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x})\right| + \left|\frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})\right| \geqslant w_{l,1}(\mathbf{x}) + w_{l,2}(\mathbf{x})\right] \\
\leqslant &\mathbb{P}\left[\left|f(\mathbf{x}) - \frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x})\right| \geqslant w_{l,1}(\mathbf{x})\right] + \mathbb{P}\left[\left|\frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})\right| \geqslant w_{l,2}(\mathbf{x})\right],
\end{aligned} \tag{43}$$

where the last inequality is from union bound.

In the following, we try to bound the above two terms, respectively.

i) Recall that each user u is associated with a local reward function $f_u \sim \mathcal{GP}(f(\cdot), k(\cdot))$. Hence,

$$f_u(\mathbf{x}) \sim \mathcal{N}(f(\mathbf{x}), k(\mathbf{x},\mathbf{x})), \quad \forall \mathbf{x} \in \mathcal{D}. \tag{44}$$

Note that U_l is a set of $\lceil 2^{\alpha l}\rceil$ independently sampled random users. Then, we have

$$\frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x}) \sim \mathcal{N}\left(f(\mathbf{x}), \frac{k(\mathbf{x},\mathbf{x})}{|U_l|}\right), \quad \forall \mathbf{x} \in \mathcal{D}.$$

Combining the concentration inequality for Gaussian random variables, we have

$$\mathbb{P}\left[\left|\frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x}) - f(\mathbf{x})\right| \geqslant w_{l,1}(\mathbf{x})\right] \leqslant 2\exp\left(-\frac{|U_l| w_{l,1}^2(\mathbf{x})}{2k(\mathbf{x},\mathbf{x}))}\right) = 2\beta. \tag{45}$$

ii) Then, we want to bound the second term in Eq. (43):

$$\begin{aligned}
&\mathbb{P}\left[\left|\frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})\right| \geqslant w_{l,2}(\mathbf{x})\right] \\
= &\sum_{\Lambda} \mathbb{P}[\Lambda = \{\mathbf{y}_{l,u}\}_{u\in U_l}] \cdot \mathbb{P}\left[\left|\frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})\right| \geqslant w_{l,2}(\mathbf{x}) \middle| \{\mathbf{y}_{l,u}\}_{u\in U_l}\right],
\end{aligned}$$

where $\mathbf{y}_{l,u} = [y_{u,t_l+1}, \ldots, y_{u,t_l+T_l}]^\top$ denotes the realization of the local reward observations at user u in the l-th phase. According to our assumption, the participant user u is associated with a local reward function f_u sampled from Gaussian Process $\mathcal{GP}(f(\cdot), k(\cdot,\cdot))$. Given the points $\mathbf{X}_{T_l} = [x_{t_l+1}^\top, \ldots, \mathbf{x}_{t_l+T_l}^\top]^\top$ in $\mathcal{D}$, the corresponding vector of local rewards $\mathbf{y}_{l,u} = [y_{u,t_l+1}, \ldots, y_{u,t_l+T_l}]^\top$ has the multivariate Gaussian distribution $\mathcal{N}(f(\mathbf{X}_{T_l}), (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda\mathbf{I}))$ where $f(\mathbf{X}_{T_l}) = [f(\mathbf{x}_{t_l+1}), \cdots, f(\mathbf{x}_{t_l+T_l})]^\top$ and $\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} = [k(\mathbf{x},\mathbf{x}')]_{\mathbf{x},\mathbf{x}'\in\mathbf{X}_{T_l}}$ is the kernel matrix for the T_l selected actions in the l-th phase. Due to the properties of GPs, we have that $\mathbf{y}_{l,u}$ and $f_u(\mathbf{x})$ are jointly Gaussian given $\mathbf{X}_{T_l}$:

$$\begin{bmatrix} f_u(\mathbf{x}) \\ \mathbf{y}_{l,u} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} f(\mathbf{x}) \\ f(\mathbf{X}_{T_l}) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x},\mathbf{x}) & \mathbf{k}(\mathbf{x},\mathbf{X}_{T_l})^\top \\ \mathbf{k}(\mathbf{x},\mathbf{X}_{T_l}) & \mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda\mathbf{I} \end{bmatrix}\right), \tag{46}$$

where $k(\mathbf{x},\mathbf{X}_{T_l}) = [k(\mathbf{x},\mathbf{x}_{t_l+1}), \ldots, k(\mathbf{x},\mathbf{x}_{t_l+T_l})]^\top$. According to the basic formula for conditional distributions of Gaussian random vectors (see [30, Appendix A.2] or [19, Proposition 3.2]), we have that conditioned on $\mathbf{y}_{l,u}$ (corresponding to the points $\mathbf{X}_{T_l}$), the following holds:

$$f_u(\mathbf{x})|\mathbf{y}_{l,u} \sim \mathcal{N}(m_u(\mathbf{x}), \sigma_{T_l}^2(\mathbf{x})),$$

where we have

$$m_u(\mathbf{x}) \triangleq f(\mathbf{x}) + \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l})^\top (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda \mathbf{I})^{-1} (\mathbf{y}_{l,u} - f(\mathbf{X}_{T_l})), \tag{47}$$

$$\sigma^2_{T_l}(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l})^\top (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda \mathbf{I})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l}). \tag{48}$$

Note that we sample the participants U_l independently and that the local reward noise is also independent across participants. Then, we have the following result:

$$\left(\frac{1}{|U_l|} \sum_{u \in U_l} f_u(\mathbf{x}) \right) \Big| \ \{\mathbf{y}_{l,u}\}_{u \in U_l} = \frac{1}{|U_l|} \sum_{u \in U_l} (f_u(\mathbf{x}) \mid \mathbf{y}_{l,u}) \sim \mathcal{N}\left(\frac{1}{|U_l|} \sum_{u \in U_l} m_u(\mathbf{x}), \frac{\sigma^2_{T_l}(\mathbf{x})}{|U_l|} \right).$$

Combining the Gaussian concentration inequality, we have the following result

$$\mathbb{P}\left[\left| \frac{1}{|U_l|} \sum_{u \in U_l} f_u(\mathbf{x}) - \frac{1}{|U_l|} \sum_{u \in U_l} m_u(\mathbf{x}) \right| \geqslant \sqrt{\frac{2\sigma^2_{T_l}(\mathbf{x}) \log(1/\beta)}{|U_l|}} \Bigg| \{\mathbf{y}_{l,u}\}_{u \in U_l} \right] \leqslant 2\beta. \tag{49}$$

From Lemma B.1, we have the following equation:

$$\frac{1}{|U_l|} \sum_{u \in U_l} \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l})^\top (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda \mathbf{I})^{-1} \mathbf{y}_{l,u} = \mathbf{k}(\mathbf{x}, \mathbf{A}_h)^\top (\mathbf{K}_{\mathbf{A}_h\mathbf{A}_h} + \lambda \mathbf{W}_h^{-1})^{-1} \bar{\mathbf{y}}_l = \bar{\mu}_l(\mathbf{x}), \tag{50}$$

which implies

$$\frac{1}{|U_l|} \sum_{u \in U_l} m_u(\mathbf{x}) = \bar{\mu}_l(\mathbf{x}) + f(\mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l})^\top (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda \mathbf{I})^{-1} f(\mathbf{X}_{T_l}). \tag{51}$$

Then, the gap between the average local function $\frac{1}{|U_l|} \sum_{u \in U_l} f_u(\cdot)$ and the estimator $\bar{\mu}_l(\cdot)$ satisfies

$$\begin{aligned}
&\left| \frac{1}{|U_l|} \sum_{u \in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x}) \right| \\
\leqslant &\left| \frac{1}{|U_l|} \sum_{u \in U_l} f_u(\mathbf{x}) - \frac{1}{|U_l|} \sum_{u \in U_l} m_u(\mathbf{x}) \right| + \left| f(\mathbf{x}) - \mathbf{k}(\mathbf{x}, \mathbf{X}_{T_l})^\top (\mathbf{K}_{\mathbf{X}_{T_l}\mathbf{X}_{T_l}} + \lambda \mathbf{I})^{-1} f(\mathbf{X}_{T_l}) \right| \\
\overset{(a)}{\leqslant} &\left| \frac{1}{|U_l|} \sum_{u \in U_l} f_u(\mathbf{x}) - \frac{1}{|U_l|} \sum_{u \in U_l} m_u(\mathbf{x}) \right| + B\sigma_{T_l}(\mathbf{x}),
\end{aligned} \tag{52}$$

where (a) is from Lemma B.4. Combining the result in Eq. (49), we have

$$\begin{aligned}
&\mathbb{P}\left[\left| \frac{1}{|U_l|} \sum_{u \in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x}) \right| \geqslant w_{l,2}(\mathbf{x}) \Bigg| \{\mathbf{y}_{l,u}\}_{u \in U_l} \right] \\
\leqslant &\mathbb{P}\left[\left| \frac{1}{|U_l|} \sum_{u \in U_l} f_u(\mathbf{x}) - \frac{1}{|U_l|} \sum_{u \in U_l} m_u(\mathbf{x}) \right| + B\sigma_{T_l}(x_0) \geqslant w_{l,2}(\mathbf{x}) \Bigg| \{\mathbf{y}_{l,u}\}_{u \in U_l} \right] \\
\overset{(a)}{=} &\mathbb{P}\left[\left| \frac{1}{|U_l|} \sum_{u \in U_l} f_u(\mathbf{x}) - \frac{1}{|U_l|} \sum_{u \in U_l} m_u(\mathbf{x}) \right| \geqslant \sqrt{\frac{2\sigma^2_{T_l}(\mathbf{x}) \log(1/\beta)}{|U_l|}} \Bigg| \{\mathbf{y}_{l,u}\}_{u \in U_l} \right] \leqslant 2\beta,
\end{aligned} \tag{53}$$

where (a) is from $\sigma^2_{T_l}(\mathbf{x}) = \Sigma^2_{H_l}(\mathbf{x})$ according to Lemma B.1. Therefore, we derive the desired result:

$$\begin{aligned}
&\mathbb{P}\left[\left|\frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})\right| \geqslant w_{l,2}(\mathbf{x})\right] \\
&= \sum_{\Lambda} \mathbb{P}[\Lambda = \{\mathbf{y}_{l,u}\}_{u\in U_l}] \cdot \mathbb{P}\left[\left|\frac{1}{|U_l|}\sum_{u\in U_l} f_u(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})\right| \geqslant w_{l,2}(\mathbf{x}) \middle| \{\mathbf{y}_{l,u}\}_{u\in U_l}\right] \\
&\leqslant \sum_{\Lambda} \mathbb{P}[\Lambda = \{\mathbf{y}_{l,u}\}_{u\in U_l}] \cdot 2\beta = 2\beta.
\end{aligned} \tag{54}$$

□

To prove Theorem 5.1, we first present three main conclusions when the concentration inequality in Theorem C.1 holds, then get an upper bound for the regret incurred in a particular phase l with high probability, and finally sum up the regret over all phases.

Define a "good" event when Eq. (41) holds in the l-th phase as:

$$\mathcal{E}_l \triangleq \{\forall \mathbf{x} \in \mathcal{D}_l, |f(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})| \leqslant w_l(\mathbf{x})\}.$$

We have $\mathbb{P}[\mathcal{E}_l] \geqslant 1 - 4|\mathcal{D}|\beta$ via the union bound. Then, under event $\mathcal{E}_l$ in the l-th phase, we have the following three observations:

1. For any optimal action $\mathbf{x}^* \in \operatorname{argmax}_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x})$, if $\mathbf{x}^* \in \mathcal{D}_l$, then $\mathbf{x}^* \in \mathcal{D}_{l+1}$.
2. Let $f^* = \max_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x})$. Supposed that $x^* \in \mathcal{D}_l$. For any $\mathbf{x} \in \mathcal{D}_{l+1}$, its reward gap from the optimal reward is bounded by $4\max_{\mathbf{x}\in\mathcal{D}_l} w_l(\mathbf{x})$, i.e.,
$$f^* - f(\mathbf{x}) \leqslant 4\max_{\mathbf{x}\in\mathcal{D}_l} w_l(\mathbf{x}).$$
3. The confidence width function satisfies
$$\max_{\mathbf{x}\in\mathcal{D}_l} w_l(\mathbf{x}) \leqslant \sqrt{\frac{2\kappa^2\log(1/\beta)}{|U_l|}} + \sqrt{\frac{4\sigma^2C^2\gamma_{T_l}\log(1/\beta)}{T_l|U_l|}} + \sqrt{\frac{2\sigma^2B^2C^2\gamma_{T_l}}{T_l}}.$$

Proof. **Observation 1:** Let $\mathbf{b} \in \operatorname{argmax}_{\mathbf{x}\in\mathcal{D}_l}(\bar{\mu}_l(\mathbf{x}) - w_l(\mathbf{x}))$. Then under event $\mathcal{E}_l$, we have

$$\bar{\mu}_l(\mathbf{x}^*) + w_l(\mathbf{x}^*) \geqslant f(\mathbf{x}^*) \geqslant f(\mathbf{b}) \geqslant \bar{\mu}_l(\mathbf{b}) - w_l(\mathbf{b}), \tag{55}$$

which indicates $\mathbf{x}^* \in \mathcal{D}_{l+1}$ according to Eq. (10).

Observation 2: For any $\mathbf{x} \in \mathcal{D}_{l+1}$, we have $\mathbf{x} \in \mathcal{D}_l$ and

$$\bar{\mu}_l(\mathbf{x}) + w_l(\mathbf{x}) \geqslant \bar{\mu}_l(\mathbf{b}) - w_l(\mathbf{b}) \geqslant \bar{\mu}_l(\mathbf{x}^*) - w_l(\mathbf{x}^*). \tag{56}$$

Then, we have the regret of choosing any action $\mathbf{x} \in \mathcal{D}_{l+1}$ satisfying

$$\begin{aligned}
f(\mathbf{x}^*) - f(\mathbf{x}) &\overset{(a)}{\leqslant} \bar{\mu}_l(\mathbf{x}^*) + w_l(\mathbf{x}^*) - \bar{\mu}_l(\mathbf{x}) + w_l(\mathbf{x}) \\
&\overset{(b)}{\leqslant} 2(w_l(\mathbf{x}) + w_l(\mathbf{x}^*)) \\
&\leqslant 4\max_{\mathbf{x}\in\mathcal{D}_l} w_l(\mathbf{x}),
\end{aligned} \tag{57}$$

where (a) holds under event $\mathcal{E}_l$ and the second inequality (b) is from Eq. (56). Then, we derive Observation 2.

Observation 3: Based on the result in Lemma B.3, we have

$$\begin{aligned}\max_{\mathbf{x}\in\mathcal{D}_l} w_l(\mathbf{x}) &= \max_{\mathbf{x}\in\mathcal{D}_l}\left(\sqrt{\frac{2k(\mathbf{x},\mathbf{x})\log(1/\beta)}{|U_l|}}+\Sigma_{H_l}(\mathbf{x})\left(\sqrt{\frac{2\log(1/\beta)}{|U_l|}}+B\right)\right)\\ &\leqslant \sqrt{\frac{2\kappa^2\log(1/\beta)}{|U_l|}}+\max_{\mathbf{x}\in\mathcal{D}_l}\Sigma_{H_l}(\mathbf{x})\left(\sqrt{\frac{2\log(1/\beta)}{|U_l|}}+B\right)\\ &\leqslant \sqrt{\frac{2\kappa^2\log(1/\beta)}{|U_l|}}+\sqrt{\frac{4\lambda C^2\gamma_{T_l}\log(1/\beta)}{T_l|U_l|}}+\sqrt{\frac{2\lambda B^2C^2\gamma_{T_l}}{T_l}}\\ &= \sqrt{\frac{2\kappa^2\log(1/\beta)}{|U_l|}}+\sqrt{\frac{4\sigma^2C^2\gamma_{T_l}\log(1/\beta)}{T_l|U_l|}}+\sqrt{\frac{2\sigma^2B^2C^2\gamma_{T_l}}{T_l}}.\end{aligned} \tag{58}$$

□

Then, we are ready to prove Theorem 5.1.

Proof of Theorem 5.1. Let the regret in the l-th phase be $r_l \triangleq \sum_{t\in\mathcal{T}_l}(\max_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x}) - f(\mathbf{x}_t))$. For any $l \geqslant 2$, we assume event $\mathcal{E}_{l-1}$ holds. Then, we have the following result

$$\begin{aligned} r_l &= \sum_{t\in\mathcal{T}_l}(\max_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x}) - f(\mathbf{x}))\\ &\leqslant \sum_{t\in\mathcal{T}_l} 4\max_{\mathbf{x}\in\mathcal{D}_{l-1}} w_{l-1}(\mathbf{x})\\ &\leqslant 4T_l\max_{\mathbf{x}\in\mathcal{D}_{l-1}} w_{l-1}(\mathbf{x})\\ &\leqslant 4T_l\left(\sqrt{\frac{2\kappa^2\log(1/\beta)}{|U_{l-1}|}}+\sqrt{\frac{4\sigma^2C^2\gamma_{T_{l-1}}\log(1/\beta)}{T_{l-1}|U_{l-1}|}}+\sqrt{\frac{2\sigma^2B^2C^2\gamma_{T_{l-1}}}{T_{l-1}}}\right)\\ &\overset{(a)}{\leqslant} 4\cdot 2^{l-1}\left(\sqrt{\frac{2\kappa^2\log(1/\beta)}{2^{\alpha(l-1)}}}+\sqrt{\frac{4\sigma^2C^2\gamma_T\log(1/\beta)}{2^{(1+\alpha)(l-1)-1}}}+\sqrt{\frac{2\sigma^2B^2C^2\gamma_T}{2^{l-2}}}\right)\\ &\leqslant 4\sqrt{2\kappa^2\log(1/\beta)}\sqrt{2^{(2-\alpha)(l-1)}}+8\sigma C\sqrt{2\gamma_T\log(1/\beta)}\sqrt{2^{(1-\alpha)(l-1)}}+8\sigma BC\sqrt{\gamma_T 2^{l-1}},\end{aligned} \tag{59}$$

where (a) is from $\gamma_{T_{l-1}} \leqslant \gamma_T$ and $|U_l| \geqslant 2^{\alpha l}$.

Define $\mathcal{E}_g$ as the event where the "good" event occurs in every phase, i.e., $\mathcal{E}_g \triangleq \bigcap_{l=1}^{L}\mathcal{E}_l$. It is not difficult to obtain $\mathbb{P}[\mathcal{E}_g] \geqslant 1 - 4|\mathcal{D}|\beta L$ by applying union bound. At the same time, let R_g be the regret under event $\mathcal{E}_g$, and R_b be the regret if event $\mathcal{E}_g$ does not hold. Then, the expected total regret in T is $\mathbb{E}[R(T)] = \mathbb{P}[\mathcal{E}_g]R_g + (1-\mathbb{P}[\mathcal{E}_g])R_b$.

Under event $\mathcal{E}_g$, the regret in the l-th phase r_l satisfies Eq. (59) for any $l \geqslant 2$. Note that $r_1 \leqslant 2T_1B\kappa \leqslant 2B\kappa$ since $T_1 = 1$ and for any $\mathbf{x}\in\mathcal{D}$,

$$|f(\mathbf{x})| = |\langle f, k(\mathbf{x},\cdot)\rangle_k| \leqslant \|f\|_k\langle k(\mathbf{x},\cdot),k(\mathbf{x},\cdot)\rangle_k^{1/2} \leqslant Bk(\mathbf{x},\mathbf{x})^{1/2} \leqslant B\kappa.$$

Then, we have

$$\begin{aligned}
R_g &= \sum_{l=1}^{L} r_l \\
&\leqslant 2B\kappa + \sum_{l=2}^{L} 4\sqrt{2\kappa^2 \log(1/\beta)}\sqrt{2^{(2-\alpha)(l-1)}} \\
&+ \sum_{l=2}^{L} 8\sigma C\sqrt{2\gamma_T \log(1/\beta)}\sqrt{2^{(1-\alpha)(l-1)}} \\
&+ \sum_{l=2}^{L} 8\sigma BC\sqrt{\gamma_T 2^{l-1}} \\
&\leqslant 2B\kappa + 4\sqrt{2\kappa^2 \log(1/\beta)} \cdot 4\sqrt{2^{(L-1)(2-\alpha)}} \\
&+ 8\sigma C\sqrt{2\gamma_T \log(1/\beta)} \cdot C_1\sqrt{2^{(1-\alpha)(L-1)}} \qquad \left(C_1 = \sqrt{2^{1-\alpha}}/(\sqrt{2^{1-\alpha}} - 1)\right) \\
&+ 8\sigma BC\sqrt{\gamma_T} \cdot 4\sqrt{2^{L-1}} \\
&\leqslant 2B\kappa + 16\sqrt{2\kappa^2 \log(1/\beta)}T^{1-\alpha/2} + 8\sigma C_1 C\sqrt{2\gamma_T \log(1/\beta)T^{1-\alpha}} + 32\sigma BC\sqrt{\gamma_T T},
\end{aligned} \tag{60}$$

where the last step is due to $2^{L-1} \leqslant T$ and $L \leqslant \log(2T)$ since $\sum_{l=1}^{L-1} T_l + 1 \leqslant T$.

On the other hand, $R_b \leqslant 2B\kappa T$ since $|\max_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x}) - f(\mathbf{x})| \leqslant 2B\kappa$ for all $\mathbf{x} \in \mathcal{D}$. Choose $\beta = 1/(|\mathcal{D}|T)$ in Algorithm 1. Finally, we have the following results:

$$\begin{aligned}
&\mathbb{E}[R(T)] \\
&= \mathbb{P}[\mathcal{E}_g]R_g + (1 - \mathbb{P}[\mathcal{E}_g])R_b \\
&\leqslant R_g + 4|\mathcal{D}|\beta L \cdot 2B\kappa T \\
&\leqslant 2B\kappa + 16\sqrt{2\kappa^2 \log(1/\beta)}T^{1-\alpha/2} + 8\sigma C_1 C\sqrt{2\gamma_T \log(1/\beta)T^{1-\alpha}} + 32\sigma BC\sqrt{\gamma_T T} \\
&+ 8B\kappa|\mathcal{D}|\beta LT \\
&= 2B\kappa + 16T^{1-\alpha/2}\sqrt{2\kappa^2 \log(|\mathcal{D}|T)} + 8\sigma C_1 C\sqrt{2\gamma_T T^{1-\alpha} \log(|\mathcal{D}|T)} \\
&+ 32\sigma BC\sqrt{\gamma_T T} + 8B\kappa \log(2T) \\
&= O(T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)}) + O(\sqrt{\gamma_T T^{1-\alpha} \log(|\mathcal{D})T)} + O(\sqrt{\gamma_T T}).
\end{aligned} \tag{61}$$

□

D PROOFS FOR COMMUNICATION AND COMPUTATION RESULTS

The results regarding computation complexity and communication cost highly depend on the number of batches H_l in each phase l. Hence, we first provide the proof for Lemma 5.6.

Proof of Lemma 5.6. To bound the number of batches in the l-th phase, we follow a similar line to the proof of Lemma 4.3 in [7]. For any $1 \leqslant h \leqslant H_l$, we have

$$\begin{aligned}
T_l(\mathbf{a}_h) = \left\lfloor \frac{C^2 - 1}{\Sigma_{h-1}^2(\mathbf{a}_h)} \right\rfloor &\geqslant \frac{C^2 - 1}{\Sigma_{h-1}^2(\mathbf{a}_h)} - 1 \\
\Rightarrow \quad \Sigma_{h-1}^2(\mathbf{a}_h)(T_l(\mathbf{a}_h) + 1) &\geqslant C^2 - 1 \\
\Rightarrow \quad 2\Sigma_{h-1}^2(\mathbf{a}_h)T_l(\mathbf{a}_h) &\geqslant C^2 - 1.
\end{aligned} \tag{62}$$

Recall that we use τ_h to denote the last within-phase time index in the h-th batch. Then, summing the above inequality across all batches up to H_l, we have

$$\begin{aligned}
H_l(C^2-1) &\leqslant \sum_{h=1}^{H_l} 2\Sigma_{h-1}^2(\mathbf{a}_h) T_l(\mathbf{a}_h) \\
&\leqslant 2\sum_{h=1}^{H_l} \sum_{\tau=\tau_{h-1}+1}^{\tau_{h-1}+T_l(\mathbf{a}_h)} \Sigma_{h-1}^2(\mathbf{a}_h) \\
&= 2\sum_{h=1}^{H_l} \sum_{\tau=\tau_{h-1}+1}^{\tau_{h-1}+T_l(\mathbf{a}_h)} \frac{\Sigma_{h-1}^2(\mathbf{a}_h)}{\sigma_\tau^2(\mathbf{a}_h)} \cdot \sigma_\tau^2(\mathbf{a}_h) \\
&\overset{(a)}{\leqslant} 2\sum_{h=1}^{H_l} \sum_{\tau=\tau_{h-1}+1}^{\tau_{h-1}+T_l(\mathbf{a}_h)} C^2 \cdot \sigma_\tau^2(\mathbf{a}_h) \\
&\overset{(b)}{=} 2C^2 \sum_{h=1}^{H_l} \sum_{\tau=\tau_{h-1}+1}^{\tau_{h-1}+T_l(\mathbf{a}_h)} \sigma_\tau^2(\mathbf{x}_{t_l+\tau}) \\
&= 2C^2 \sum_{\tau=1}^{T_l} \sigma_\tau^2(\mathbf{x}_{t_l+\tau}) \\
&\overset{(c)}{\leqslant} 4\sigma^2 C^2 \gamma_{T_l},
\end{aligned} \tag{63}$$

where (a) is from Corollary B.2, (b) is based on our algorithm decision: $\mathbf{x}_{t_l+\tau} = \mathbf{a}_h$ for any $\tau \in [\tau_{h-1}+1, \tau_{h-1}+T_l(\mathbf{a}_h)]$, (c) is from Lemma A.1 where $\mathbf{X}_{T_l} = [\mathbf{x}_{t_l+1}^\top, \ldots, \mathbf{x}_{t_l+T_l}^\top]^\top$ for any phase l and $\lambda = \sigma^2$. Hence, we derive

$$H_l \leqslant \frac{4\sigma^2 C^2}{C^2-1} \gamma_{T_l}. \tag{64}$$

□

We already analyze how to derive the computation complexity for DPBE in Remark 5.7. In the following, we prove Theorem 5.8, which tells the result regarding communication cost: $O(\gamma_T T^\alpha)$.

Proof of Theorem 5.8. Note that the communicating data in each phase between participants and the agent is the local average performance $y_l^u(\mathbf{a})$ for each action $\mathbf{a}$ chosen in the corresponding batch. That is, $N_{u,l} \leqslant H_l$ for every participant u. (Here, the inequality holds when merging batches as Remark 4.1). Combining the bound of H_l in Lemma 5.6, we derive the total communication cost satisfying

$$\sum_{l=1}^{L} |U_l| H_l \leqslant \sum_{l=1}^{L} \frac{4\sigma^2 C^2}{C^2-1} \gamma_{T_l} \cdot (2^{\alpha l}+1) = O\left(\frac{\sigma^2 C^2}{C^2-1} \cdot \gamma_T T^\alpha\right), \tag{65}$$

where the last step is due to $2^{L-1} \leqslant T$ and $L \leqslant \log(2T)$ since $\sum_{l=1}^{L-1} T_l + 1 \leqslant T$. □

E DIFFERENTIALLY PRIVATE DPBE EXTENSIONS

In this section, we extend the differentially private DPBE in Section 6 to two other celebrated DP models: the local model and the shuffle model.

To begin with, we present the details of the DP-DPBE algorithm (see Algorithm 2) in the central DP model discussed in Section 6. Recall that in the central DP model, with a trusted agent, data privacy is protected by privatizing the aggregated feedback so that the output of the algorithm is indistinguishable between any two users. In a particular phase l, the aggregated feedback for each

chosen action becomes $\tilde{\mathbf{y}}_l = \bar{\mathbf{y}}_l + (\rho_1, \ldots, \rho_{H_l})$ (see Eq. (15)), where $\rho_j \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_{nc}^2)$ is the injected Gaussian noise for ensuring the required (ε, δ)-DP and is chosen according to the (high-probability) sensitivity of $\bar{\mathbf{y}}_l$. Specifically, we set the variance of the injected Gaussian noise to the following:

$$\sigma_{nc} = \frac{2\sqrt{2(\kappa^2 + \sigma^2) H_l \log(2H_l/\delta_1) \ln(1.25/\delta_2)}}{\varepsilon |U_l|}, \tag{66}$$

where $\delta_1 \in (0, \delta)$ is the probability of sensitivity concentration of $\bar{\mathbf{y}}_l$ (i.e., Eq. (77) holds with probability at least $1-\delta_1$) and $\delta_2 = \delta - \delta_1$. By accounting for privacy noise, we update the confidence width function in Eq. (17) with $\sigma_n = \sigma_{nc}\sqrt{2C^2\gamma_T}$, where C is the rare-switching parameter.

Differentially Private DPBE in the Local DP Model. In the local model, the users do not trust the agent, and thus, each is equipped with a local randomizer $\mathcal{R}$ to protect its own local reward. Let Y be the set of all possible values of the local reward. Formally, a local randomizer $\mathcal{R}$ is (ε, δ)-local differentially private (or (ε, δ)-LDP) if for any two user inputs, the probability that $\mathcal{R}$ outputs two values in Y that are not different by more than a multiplicative factor of e^{ϵ} and an additive factor of δ. To guarantee LDP, the local randomizer $\mathcal{R}$ at each user u injects Gaussian noise before sending the local reward observations out to the central agent. That is,

$$\mathcal{R}(\mathbf{y}_l^u) = \mathbf{y}_l^u + (\rho_{u,1}, \ldots, \rho_{u,H_l}), \tag{70}$$

where $\rho_{u,j} \sim \mathcal{N}(0, \sigma_{nl}^2)$ is *i.i.d.* across both users and actions and the variance σ_{nl}^2 is chosen according to the (high-probability) sensitivity of $\mathbf{y}_l^u$ (see Eq. (80)). Then, the *private* aggregated feedback for the chosen actions in the l-th phase in the local DP model becomes

$$\tilde{\mathbf{y}}_l = \frac{1}{|U_l|} \sum_{u \in U_l} \mathcal{R}(\mathbf{y}_l^u) = \frac{1}{|U_l|} \sum_{u \in U_l} \left(\mathbf{y}_l^u + (\rho_{u,1}, \ldots, \rho_{u,H_l}) \right). \tag{71}$$

We call the differentially private version of DPBE in the local DP model LDP-DPBE. Specifically, we extend the DPBE algorithm (Algorithm 2) to LDP-DPBE by employing a local randomizer $\mathcal{R}$ as in Eq. (70) at each participant in the l-th phase and then using the privately aggregated feedback in Eq. (71) to estimate the mean function $\tilde{\mu}_l(\cdot)$ in Eq. (16). The injected Gaussian noise at each participant is $\sigma_{nl} = \frac{2\sqrt{2(\kappa^2+\sigma^2) H_l \log(2H_l/\delta_1) \ln(1.25/\delta_2))}}{\varepsilon}$, where $\delta_1 \in (0, \delta)$ is the probability of sensitivity concentration of $\bar{\mathbf{y}}_l$ (i.e., Eq. (80) holds with probability at least $1 - \delta_1$) and $\delta_2 = \delta - \delta_1$. In LDP-DPBE, we update the confidence width function in Eq. (17) with $\sigma_n = \sqrt{\frac{2C^2\sigma_{nl}^2\gamma_T}{|U_l|}}$, where C is the rare-switching parameter.

Differentially Private DPBE in the Shuffle DP Model. While local DP provides a more stringent privacy guarantee, it usually incurs larger regret cost [42]. The shuffle model is recently proposed to achieve a better tradeoff between regret and privacy [11]. In the shuffle model, between the users and the agent, there exists a shuffler that permutes the local feedback from the participants before they are observed by the agent so that the agent cannot distinguish between two users' feedback. Thus, an additional layer of randomness is introduced via shuffling, which can often be easily implemented using Cryptographic primitives (e.g., mixnets) due to its simple operation [2]. Specifically, the shuffle DP model consists of three components: a local randomizer $\mathcal{R}$ at each user side, a shuffler $\mathcal{S}$ between the users and the agent, and an analyzer $\mathcal{A}$ at the agent side. Let $\mathcal{U}_T \triangleq (U_l, \cdots, U_l)$ be the participants throughout the T rounds. Define the (composite) mechanism $\mathcal{M}_s(\mathcal{U}_T) \triangleq ((\mathcal{S} \circ \mathcal{R})(U_1), (\mathcal{S} \circ \mathcal{R})(U_2), \ldots, (\mathcal{S} \circ \mathcal{R})(U_L))$, where $(\mathcal{S} \circ \mathcal{R})(U_l) \triangleq \mathcal{S}(\{\mathcal{R}(\mathbf{y}_l^u)\}_{u \in U_l})$. Formally, We say the DP-DPBE algorithm satisfies the shuffle differential privacy (SDP) if the composite mechanism $\mathcal{M}_s$ is DP, which leads to the following formal definition.

Algorithm 2 Differentially Private Distributed Phase-then-Batch-based Elimination (DP-DPBE)

Input: $\mathcal{D} \subseteq \mathbb{R}^d$, $\alpha \in (0,1)$, $\beta \in (0,1)$, rare-switching parameter C, local noise σ^2, privacy parameters ε and δ
Initialization: $l = 1$, $\mathcal{D}_1 = \mathcal{D}$, $t_1 = 0$, and $T_1 = 1$
while $t_l < T$ **do**
  Set $\tau = 1$, $h = 0$, $\tau_1 = 0$ and $\Sigma_0^2(\mathbf{x}) = k(\mathbf{x},\mathbf{x})$, for all $\mathbf{x} \in \mathcal{D}_l$
  while $\tau \leqslant T_l$ **do**
    $h = h + 1$
    Choose

$$\mathbf{a}_h \in \underset{\mathbf{x}\in\mathcal{D}_l}{\operatorname{argmax}} \Sigma_{h-1}^2(\mathbf{x}) \tag{67}$$

    Play action $\mathbf{a}_h$ for $T_l(\mathbf{a}_h) \triangleq \lfloor (C^2-1)/\Sigma_{h-1}^2(\mathbf{a}_h) \rfloor$ times if not reaching $\min\{T, t_l + T_l\}$
    Update $\tau = \tau + T_l(\mathbf{a}_h)$, and the posterior variance $\Sigma_h^2(\cdot)$ by including $\mathbf{a}_h$ according to Eq. (7).
  end while
  Let $H_l = h$ denote the total number of batches in this phase.
  Randomly select $\lceil 2^{\alpha l} \rceil$ participants U_l
  # Operations at each participant
  for each participant $u \in U_l$ **do**
    Collect and compute local average reward for every chosen action $\mathbf{a} \in \mathbf{A}_{H_l}$:

$$y_l^u(\mathbf{a}) = \frac{1}{T_l(\mathbf{a})} \sum_{t\in\mathcal{T}_l(\mathbf{a})} y_{u,t}$$

    Send the local average reward for every chosen action $\mathbf{y}_l^u \triangleq [y_l^u(\mathbf{a})]_{\mathbf{a}\in\mathbf{A}_{H_l}}$ to the agent
  end for
  Aggregate local observations for each chosen action $\mathbf{a} \in \mathbf{A}_{H_l}$:

$$y_l(\mathbf{a}) = \frac{1}{|U_l|} \sum_{u\in U_l} y_l^u(\mathbf{a})$$

  Let $\bar{\mathbf{y}}_l = [y_l(\mathbf{a}_1), \ldots, y_l(\mathbf{a}_{H_l})]$ and

$$\tilde{\mathbf{y}}_l = \bar{\mathbf{y}}_l + (\rho_1, \ldots, \rho_{H_l}),$$

  where $\rho_j \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_{nc}^2)$ and σ_{nc} is specified in Eq. (66).
  Update $\tilde{\mu}_l(\cdot)$:

$$\tilde{\mu}_l(\mathbf{x}) \triangleq \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \tilde{\mathbf{y}}_l \tag{68}$$

  Eliminate low-rewarding actions from $\mathcal{D}_l$ based on the confidence width function $\tilde{w}_l(\cdot)$ in Eq. (17) with $\sigma_n = \sigma_{nc}\sqrt{2C^2\gamma_T}$:

$$\mathcal{D}_{l+1} = \left\{ \mathbf{x} \in \mathcal{D}_l : \tilde{\mu}_l(\mathbf{x}) + \tilde{w}_l(\mathbf{x}) \geqslant \max_{\mathbf{b}\in\mathcal{D}_l} (\tilde{\mu}_l(\mathbf{b}) - \tilde{w}_l(\mathbf{b})) \right\}. \tag{69}$$

  $T_{l+1} = 2T_l$, $t = t + T_l$; $l = l + 1$
end while

Definition E.1. (Shuffle Differential Privacy (SDP)). For any $\varepsilon \geqslant 0$ and $\delta \in [0,1]$, the DP-DPBE is (ε,δ)-shuffle differential privacy (or (ε,δ)-SDP) if for any pair $\mathcal{U}_T$ and $\mathcal{U}_T'$ that differ by one user,

and for any $Z \in Range(\mathcal{M}_s)$[5]:

$$\mathbb{P}[\mathcal{M}_s(\mathcal{U}_T) \in Z] \leqslant e^{\varepsilon}\mathbb{P}[\mathcal{M}_s(\mathcal{U}_T^{'}) \in Z] + \delta. \tag{72}$$

In our case, we apply a shuffle model to the feedback from participants of every particular phase. That is, the *private* aggregated feedback for the chosen actions in the l-th phase in the shuffle DP model becomes

$$\tilde{\mathbf{y}}_l = \mathcal{A}\left(\mathcal{S}\left(\{\mathcal{R}(\mathbf{y}_l^u\}_{u\in U_l})\right)\right), \tag{73}$$

where the local randomizer injects a sub-Gaussian noise with variance σ_{ns}^2, which is *i.i.d.* across both users and actions. Thanks to our phase-then-batch strategy, the recently proposed vector summation protocol [10] can be extended to our algorithm as [22]. We present the concrete pseudocodes of $\mathcal{R}$, $\mathcal{S}$, and $\mathcal{A}$ in Algorithm 3.

We call the differentially private version of DPBE in the shuffle model SDP-DPBE, which is extended from DP-DPBE by using the privately aggregated feedback in Eq. (73), where $\mathcal{R}$, $\mathcal{S}$, and $\mathcal{A}$ are specified in Algorithm 3. For any $\delta_1 \in (0, \delta)$, let $\Delta \triangleq B\kappa\sqrt{H_l} + \sqrt{2(\kappa^2+\sigma^2)H_l \log(2H_l/\delta_1)}$. It is not difficult to show that $\|\mathbf{y}_l^u\|_2 \leqslant \Delta$ with probability at least $1-\delta_1$. SDP-DPBE employs Algorithm 3 in each phase l with input $\{\mathbf{y}_l^u\}_{u\in U_l}$, Δ, and privacy parameters ε and $\delta_2 = \delta - \delta_1$. According to [22], the introduced error for privacy is sub-Gaussian with variance $\sigma_{ns}^2 = O\left(\frac{H_l(\kappa^2+\sigma^2)\log(H_l/\delta_1)\ln(H_l/\delta_2)^2}{\varepsilon^2|U_l|^2}\right)$. In SDP-DPBE, we update the confidence width function in Eq. (17) with $\sigma_n = \sigma_{ns}\sqrt{2C^2\gamma_T}$, where C is the rare-switching parameter.

E.1 Performance Guarantee

For the DP-DPBE algorithm incorporated with the above local and shuffle DP models, we provide the DP guarantee and regret in the following.

Theorem E.2 (DP guarantee). *Under Assumptions 1, 2, 3, and 4, for any $\varepsilon > 0$ and $\delta \in (0,1)$,*

i) LDP-DPBE *guarantees (ε, δ)-LDP;*
ii) SDP-DPBE *guarantees (ε, δ)-SDP.*

We achieve the above LDP guarantee of i) directly by employing the Gaussian mechanism given the (high-probability) sensitivity of $\mathbf{y}_l^u$. In the shuffle model, we follow the shuffle protocol for each phase in [22] and derive the corresponding SDP guarantee from Theorem A.2 therein.

From the above results, we derive that compared to the local model the shuffle model injects much less noise (σ_{ns}^2 vs. σ_{nl}^2) without requiring a trusted agent. In the following, we present the regret performance of DP-DPBE in these two DP models.

Theorem E.3 (LDP-DPBE). *Under Assumptions 1, 2, and 3, the* LDP-DPBE *algorithm with $\beta = \frac{1}{|\mathcal{D}|T}$ achieves the following expected regret:*

$$\mathbb{E}[R(T)] = O(T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)}) + O\left(\frac{\ln(1/\delta)\gamma_T T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)}}{\varepsilon}\right). \tag{75}$$

Theorem E.4 (SDP-DPBE). *Under Assumptions 1, 2, and 3, the* SDP-DPBE *algorithm with $\beta = \frac{1}{|\mathcal{D}|T}$ achieves the following expected regret:*

$$\mathbb{E}[R(T)] = O(T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)}) + O\left(\frac{\ln^{3/2}(\gamma_T/\delta)\gamma_T T^{1-\alpha}\sqrt{\log(|\mathcal{D}|T)}}{\varepsilon}\right). \tag{76}$$

[5] $Rang(\mathcal{M})$ denotes the range of the output of the mechanism $\mathcal{M}$.

Algorithm 3 $\mathcal{M}$: Shuffle Protocol for a Set of Vectors with Users U [22]

Input: $\{\mathbf{y}^u\}_{u\in U}$, where each $\mathbf{y}^u \in \mathbb{R}^s$, $\|\mathbf{y}^u\|_2 \leqslant \Delta$, privacy parameters $\varepsilon, \delta_2 \in (0,1)$

Let

$$\begin{cases} \widehat{\varepsilon} = \frac{\varepsilon}{18\sqrt{\log(2/\delta_2)}} \\ g \triangleq \max\{\widehat{\varepsilon}\sqrt{|U|}/(6\sqrt{5\ln((4s)/\delta_2)}), \sqrt{s}, 10\} \\ b \triangleq \lceil \frac{180 g^2 \ln(4s/\delta_2)}{\widehat{\varepsilon}^2 |U|} \rceil \\ p \triangleq \frac{90 g^2 \ln(4s/\delta_2)}{b\widehat{\varepsilon}^2 |U|} \end{cases} \tag{74}$$

// Local Randomizer

function $\mathcal{R}(\mathbf{y}^u)$

for coordinate $j \in [s]$ **do**

Shift data to enforce non-negativity: $w_{u,j} = (\mathbf{y}^u)_j + \Delta, \forall u \in U$

//randomizer for each entry

Set $\bar{w}_{u,j} \leftarrow \lfloor w_{u,j} g/(2\Delta) \rfloor$ //$\max |(\mathbf{y}^u)_j + \Delta| \leqslant 2\Delta$

Sample rounding value $\gamma_1 \sim \mathbf{Ber}(w_{u,j} g/(2\Delta) - \bar{w}_{u,j})$

Sample privacy noise value $\gamma_2 \sim \mathbf{Bin}(b, p)$

Let ϕ_j^u be a multi-set of $(g+b)$ bits associated with the j-th coordinate of user u, where ϕ_j^u consists of $\bar{w}_{u,j} + \gamma_1 + \gamma_2$ copies of 1 and $g + b - (\bar{w}_{i,j} + \gamma_1 + \gamma_2)$ copies of 0

end for

Report $\{(j, \phi_j^u)\}_{j\in[s]}$ to the shuffler

end function

// Shuffler

function $\mathcal{S}(\{(j, \boldsymbol{\phi}_j)\}_{j\in[s]})$ //$\boldsymbol{\phi}_j = (\phi_j^u)_{u\in U}$

for each coordinate $j \in [s]$ **do**

Shuffle and output all $(g+b)|U|$ bits in $\boldsymbol{\phi}_j$

end for

end function

// Analyzer

function $\mathcal{A}(\mathcal{S}(\{(j, \boldsymbol{\phi}_j)\}_{j\in[s]})$

for coordinate $j \in [s]$ **do**

Compute $z_j \leftarrow \frac{2\Delta}{g|U|}((\sum_{i=1}^{(g+b)|U|}(\boldsymbol{\phi}_j)_i) - b|U|p)$ // $(\boldsymbol{\phi}_j)_i$ denotes the i-th bit in $\boldsymbol{\phi}_j$

Re-center: $o_j \leftarrow z_j - \Delta$

end for

Output the estimator of vector average $o = (o_j)_{j\in[s]}$

end function

We omit the proofs for the above two theorems because they can be derived by directly replacing σ_n of the central model with $\sigma_n = \sqrt{\frac{2C^2\sigma_{nl}^2\gamma_T}{|U_l|}}$ of the local model and $\sigma_n = \sigma_{ns}\sqrt{2C^2\gamma_T}$ of the shuffle model. See Appendix E.3.

E.2 Proofs for DP Guarantees

Before providing the DP guarantee of the DPBE algorithm in the three DP models, we first show the ℓ_2 sensitivity of $\bar{\mathbf{y}}_l$, which is a key parameter to decide the Gaussian noise.

Lemma E.5. *Let $\mathcal{U}_T, \mathcal{U}_T' \subseteq \mathcal{U}$ be two sets of participants in* DPBE *differing on a single user that is participating in the l-th phase, and let $\bar{\mathbf{y}}_l$ and $\bar{\mathbf{y}}_l'$ be the corresponding average local reward. For any*

Table 6. Regret of DP-DPBE in Different DP Models

Algorithms	Regret
DPBE	$O(T^{1-\alpha/2}\sqrt{\log(\lvert\mathcal{D}\rvert T)})$
CDP-DPBE	$O(T^{1-\alpha/2}\sqrt{\log(\lvert\mathcal{D}\rvert T)}) + O\left(\frac{\ln(1/\delta)\gamma_T T^{1-\alpha}\sqrt{\log(\lvert\mathcal{D}\rvert T)}}{\varepsilon}\right)$
LDP-DPBE	$O(T^{1-\alpha/2}\sqrt{\log(\lvert\mathcal{D}\rvert T)}) + O\left(\frac{\ln(1/\delta)\gamma_T T^{1-\alpha/2}\sqrt{\log(\lvert\mathcal{D}\rvert T)}}{\varepsilon}\right)$
SDP-DPBE	$O(T^{1-\alpha/2}\sqrt{\log(\lvert\mathcal{D}\rvert T)}) + O\left(\frac{\ln^{3/2}(\gamma_T/\delta)\gamma_T T^{1-\alpha}\sqrt{\log(\lvert\mathcal{D}\rvert T)}}{\varepsilon}\right)$

Notes: CDP-DPBE, LDP-DPBE, and SDP-DPBE represent the DP-DPBE algorithm in the central, local, and shuffle models, respectively, which guarantee (ε, δ)-DP, (ε, δ)-LDP, and (ε, δ)-SDP, respectively.

$\delta_1 \in (0, 1)$, we have that with probability at least $1 - \delta_1$, the maximal ℓ_2 distance between $\bar{\mathbf{y}}_l$ and $\bar{\mathbf{y}}'_l$ is bounded by

$$\max |\bar{\mathbf{y}}'_l - \bar{\mathbf{y}}_l| \leqslant 2\frac{\sqrt{(\kappa^2 + \sigma^2)H_l \log(2H_l/\delta_1)}}{|U_l|}, \tag{77}$$

where H_l denotes the dimension of $\bar{\mathbf{y}}_l$ and σ^2 is the variance of the noisy observations.

Proof. Let U_l, U'_l be the sets of participating users in l-th phase corresponding to $\mathcal{U}_T$ and $\mathcal{U}'_T$ respectively. We have $|U_l| = |U'_l|$ and the maximal ℓ_2 distance between $\bar{\mathbf{y}}_l, \bar{\mathbf{y}}'_l$ is the following:

$$\begin{aligned}\max |\bar{\mathbf{y}}'_l - \bar{\mathbf{y}}_l| &= \max_{\mathcal{U}_T, \mathcal{U}'_T} \left\| \frac{1}{|U_l|} \sum_{u \in U'_l} \mathbf{y}_l^u - \frac{1}{|U_l|} \sum_{u \in U_l} \mathbf{y}_l^u \right\|_2 \\ &= \frac{1}{|U_l|} \max_{u, u' \in \mathcal{U}} \|\mathbf{y}_l^{u'} - \mathbf{y}_l^u\|_2.\end{aligned} \tag{78}$$

For any chosen action $\mathbf{a} \in \mathbf{A}_{H_l}$, we have the following result:

$$\begin{aligned}|y_l^{u'}(\mathbf{a}) - y_l^u(\mathbf{a})| &= \left| \frac{1}{T_l(\mathbf{a})} \sum_{t \in \mathcal{T}_l(\mathbf{a})} y_{u',t} - \frac{1}{T_l(\mathbf{a})} \sum_{t \in \mathcal{T}_l(\mathbf{a})} y_{u,t} \right| \\ &= \left| \frac{1}{T_l(\mathbf{a})} \sum_{t \in \mathcal{T}_l(\mathbf{a})} (y_{u',t} - y_{u,t}) \right| \\ &= \left| \frac{1}{T_l(\mathbf{a})} \sum_{t \in \mathcal{T}_l(\mathbf{a})} (f_{u'}(\mathbf{x}_t) + \eta_{u',t} - f_u(\mathbf{x}_t) - \eta_{u,t}) \right| \\ &\leqslant \frac{1}{T_l(\mathbf{a})} \sum_{t \in \mathcal{T}_l(\mathbf{a})} \left| f_{u'}(\mathbf{x}_t) + \eta_{u',t} - f_u(\mathbf{x}_t) - \eta_{u,t} \right|.\end{aligned}$$

Note that $f_u(\mathbf{x}) \sim \mathcal{N}(f(\mathbf{x}), k(\mathbf{x}, \mathbf{x}))$, $\eta_{u,t} \sim \mathcal{N}(0, \sigma^2)$, and the participating users are independent from each other. We have $(f_{u'}(\mathbf{x}_t) + \eta_{u',t} - f_u(\mathbf{x}_t) - \eta_{u,t}) \sim \mathcal{N}(0, 2(k(\mathbf{x}_t, \mathbf{x}_t) + \sigma^2))$. According to the concentration property of Gaussian distribution, we have with probability at least $1 - \delta_1$,

$$|f_{u'}(\mathbf{x}_t) + \eta_{u',t} - f_u(\mathbf{x}_t) + \eta_{u,t})| \leqslant 2\sqrt{(k(\mathbf{x}_t, \mathbf{x}_t) + \sigma^2)\log(2/\delta_1)} \leqslant 2\sqrt{(\kappa^2 + \sigma^2)\log(2/\delta_1)}, \tag{79}$$

which results in $|y_l^{u'}(\mathbf{a}) - y_l^u(\mathbf{a})| \leqslant 2\sqrt{(\kappa^2+\sigma^2)\log(2/\delta_1)}$ for any particular $\mathbf{a} \in \mathbf{A}_{H_l}$ with probability at least $1-\delta_1$. By substituting the above result into Eq. (78) and applying union bound, we have that with probability at least $1-\delta_1$, the following is satisfied:

$$\max_{u,u' \in \mathcal{U}} \|\mathbf{y}_l^{u'} - \mathbf{y}_l^u\|_2 \leqslant 2\sqrt{H_l(\kappa^2+\sigma^2)\log(2H_l/\delta_1)}, \tag{80}$$

and then with probability at least $1-\delta_1$, the ℓ_2 distance between $\bar{\mathbf{y}}_l$ and $\bar{\mathbf{y}}_l'$ is bounded by

$$\max|\bar{\mathbf{y}}_l' - \bar{\mathbf{y}}_l| \leqslant \frac{\max_{u,u' \in \mathcal{U}} \|\mathbf{y}_l^{u'} - \mathbf{y}_l^u\|_2}{|U_l|} \leqslant \frac{2\sqrt{H_l(\kappa^2+\sigma^2)\log(2H_l/\delta_1)}}{|U_l|}, \tag{81}$$

where the last step is because H_l is the dimension of $\mathbf{y}_l^u$ and also the number of actions in $\mathbf{A}_{H_l}$. □

For both the central model and the local model, we employ the Gaussian mechanism in the differential privacy literature, which is described in the following.

Theorem E.6. *(Gaussian Mechanism [17]). Given any vector-valued function*[6] $f : \mathcal{U}^* \to \mathbb{R}^s$, *define* $\Delta_2 \triangleq \max_{\mathcal{U}_1,\mathcal{U}_2' \subseteq \mathcal{U}} \|f(\mathcal{U}_1) - f(\mathcal{U}_2)\|_2$. *Let* $\sigma = \Delta_2\sqrt{2\ln(1.25/\delta)}/\varepsilon$. *The Gaussian mechanism, which adds independently drawn random noise from* $\mathcal{N}(0,\sigma^2)$ *to each output of* $f(\cdot)$, *i.e. returning* $f(\mathcal{U}) + (\rho_1,\ldots,\rho_s)$ *with* $\rho_j \overset{i.i.d.}{\sim} \mathcal{N}(0,\sigma^2)$, *ensures* (ε,δ)*-DP.*

Proof of Theorem 6.2. Let E denote the event that Eq. (77) holds, and thus, $\mathbb{P}[E] \geqslant 1-\delta_1$. Let $\Delta_2 \triangleq \max|\bar{\mathbf{y}}_l' - \bar{\mathbf{y}}_l|$. If E holds, adding independently drawn noise from $\mathcal{N}\left(0, \frac{2\Delta_2^2\ln(1.25/\delta_2)}{\varepsilon}\right)$ to each element of $\bar{\mathbf{y}}_l$, i.e., returning $\bar{\mathbf{y}}_l + (\rho_1,\cdots,\rho_{H_l})$ with $\rho_j \overset{i.i.d.}{\sim} \mathcal{N}\left(0, \frac{2\Delta_2^2\ln(1.25/\delta_2)}{\varepsilon}\right)$, ensures (ε,δ_2)-DP . Specifically, the following inequality holds

$$\mathbb{P}[\mathcal{M}(\mathcal{U}_T) \in Z|E] \leqslant e^{\varepsilon}\mathcal{P}[\mathcal{M}(\mathcal{U}_T') \in Z|E] + \delta_2. \tag{82}$$

Then, we have

$$\begin{aligned}
\mathbb{P}[\mathcal{M}(\mathcal{U}_T) \in Z] &\leqslant \mathbb{P}[\mathcal{M}(\mathcal{U}_T) \in Z|E]\mathbb{P}[E] + 1 - \mathbb{P}[E] \\
&\leqslant (e^{\varepsilon}\mathcal{P}[\mathcal{M}(\mathcal{U}_T') \in Z|E] + \delta_2)\mathbb{P}[E] + \delta_1 \\
&\leqslant e^{\varepsilon}\mathcal{P}[\mathcal{M}(\mathcal{U}_T') \in Z|E]\mathbb{P}[E] + \delta_2 + \delta_1 \\
&\leqslant e^{\varepsilon}\mathcal{P}[\mathcal{M}(\mathcal{U}_T') \in Z|E]\mathbb{P}[E] + \delta_2 + \delta_1 \\
&\leqslant e^{\varepsilon}\mathcal{P}[\mathcal{M}(\mathcal{U}_T') \in Z, E] + \delta_2 + \delta_1 \\
&\leqslant e^{\varepsilon}\mathcal{P}[\mathcal{M}(\mathcal{U}_T') \in Z] + \delta,
\end{aligned} \tag{83}$$

where $\delta = \delta_1 + \delta_2$. □

Similarly, we can derive the (ε,δ)-LDP. Meanwhile, we can achieve (ε,δ)-SDP by combining the analysis in Eq. (83) and the proof for Theorem A.2 in [22].

E.3 Proof of Theorem 6.3

Following a similar line to the proof for Theorem 5.1, we first provide the key concentration inequality under `DP-DPBE` in Theorem E.7.

Theorem E.7. *For any particular phase* l, *with probability at least* $1-6\beta$, *the following holds*

$$|f(\mathbf{x}) - \tilde{\mu}_l(\mathbf{x})| \leqslant \tilde{w}_l(\mathbf{x}), \tag{84}$$

where mean function $\tilde{\mu}_l(\mathbf{x})$ *and confidence width function* $\tilde{w}_l(\mathbf{x})$ *are defined in Eq. (16) and Eq. (17).*

[6] We use the superscript * to indicate that the length could be varying.

Proof. In this proof, we will show the following concentration inequality holds for any $\mathbf{x} \in \mathcal{D}$

$$\mathbb{P}[|f(\mathbf{x}) - \tilde{\mu}_l(\mathbf{x})| \geqslant \tilde{w}_l(\mathbf{x})] \leqslant 6\beta. \tag{85}$$

Let $\boldsymbol{\rho} \triangleq (\rho_1, \ldots, \rho_{H_l})$. Note that

$$\begin{aligned}
\tilde{\mu}_l(\mathbf{x}) &= \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \tilde{\mathbf{y}}_l \\
&= \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} (\bar{y}_l + \boldsymbol{\rho}) \\
&= \bar{\mu}_l(x) + \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \boldsymbol{\rho}.
\end{aligned} \tag{86}$$

Then, we have

$$|f(\mathbf{x}) - \tilde{\mu}_l(\mathbf{x})| \leqslant |f(\mathbf{x}) - \bar{\mu}_l(\mathbf{x})| + |\mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \boldsymbol{\rho}|. \tag{87}$$

For any $\mathbf{x} \in \mathcal{D}$, we have

$$\begin{aligned}
&\mathbb{P}\left[|f(\mathbf{x}) - \tilde{\mu}_l(\mathbf{x})| \geqslant \tilde{w}_l(\mathbf{x})\right] \\
\leqslant &\mathbb{P}\left[|f(\mathbf{x}) - \bar{\mu}_l(x)| + \left|\mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \boldsymbol{\rho}\right| \geqslant w_l(\mathbf{x}) + 2C\sqrt{\gamma_T \sigma_n^2 \log(1/\beta)}\right] \\
\leqslant &\mathbb{P}\left[|f(\mathbf{x}) - \bar{\mu}_l(x)| \geqslant w_l(\mathbf{x})\right] + \mathbb{P}\left[\left|\mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \boldsymbol{\rho}\right| \geqslant 2C\sqrt{\gamma_T \sigma_n^2 \log(1/\beta)}\right] \\
\leqslant &4\beta + \mathbb{P}\left[\left|\mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \boldsymbol{\rho}\right| \geqslant 2C\sqrt{\gamma_T \sigma_n^2 \log(1/\beta)}\right],
\end{aligned} \tag{88}$$

where the first inequality is due to $\tilde{w}_l(x) = w_l(x) + \sqrt{2\sigma_n^2 \log(1/\beta)}$ from Eq. (17), the second inequality is from union bound, and the last one is from Theorem C.1. Hence, it remains to bound the second probability in Eq. (88).

Recall that $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_{H_l})$ where $\rho_j \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_{nc}^2)$. Then, $\mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \boldsymbol{\rho}$ is the sum of H_l *i.i.d.* Gaussian variables, and the total variance (denoted by σ_{sum}^2) is

$$\sigma_{\text{sum}}^2 = \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l}) \sigma_{nc}^2. \tag{89}$$

Notice that

$$
\begin{aligned}
&\mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l}) \\
=&\varphi(\mathbf{x})^\top \Phi_{H_l}^\top (\Phi_{H_l}\Phi_{H_l}^\top + \lambda \mathbf{W}_{H_l}^{-1})^{-1} (\Phi_{H_l}\Phi_{H_l}^\top + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \Phi_{H_l}\varphi(\mathbf{x}) \\
=&\varphi(\mathbf{x})^\top \Phi_{H_l}^\top \mathbf{W}_{H_l}^{1/2} (\mathbf{W}_{H_l}^{1/2}\Phi_{H_l}\Phi_{H_l}^\top \mathbf{W}_{H_l}^{1/2} + \lambda \mathbf{I})^{-1} \mathbf{W}_{H_l}^{1/2} \cdot \mathbf{W}_{H_l}^{1/2} (\mathbf{W}_{H_l}^{1/2}\Phi_{H_l}\Phi_{H_l}^\top \mathbf{W}_{H_l}^{1/2} + \lambda \mathbf{I})^{-1} \mathbf{W}_{H_l}^{1/2} \Phi_{H_l}\varphi(\mathbf{x}) \\
=&\varphi(\mathbf{x})^\top (\Phi_{H_l}^\top \mathbf{W}_{H_l}^{1/2} \mathbf{W}_{H_l}^{1/2} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \Phi_{H_l}^\top \mathbf{W}_{H_l}^{1/2} \cdot \mathbf{W}_{H_l} \cdot \mathbf{W}_{H_l}^{1/2} \Phi_{H_l} (\Phi_{H_l}^\top \mathbf{W}_{H_l}^{1/2} \mathbf{W}_{H_l}^{1/2} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x}) \\
=&\varphi(\mathbf{x})^\top (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \Phi_{H_l}^\top \mathbf{W}_{H_l}^2 \Phi_{H_l} (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x}) \\
\overset{(a)}{\leqslant}& T_l \varphi(\mathbf{x})^\top (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x}) \\
=&T_l \varphi(\mathbf{x})^\top (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \sigma_n^2 \mathbf{I}) (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x}) \\
&- \lambda T_l \varphi(\mathbf{x})^\top (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x}) \\
\leqslant& T_l \varphi(\mathbf{x})^\top (\Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x}) \\
\overset{(b)}{=}& T_l \varphi(\mathbf{x})^\top (\Phi_{\tau_{H_l}}^\top \Phi_{\tau_{H_l}} + \lambda \mathbf{I})^{-1} \varphi(\mathbf{x}) \\
\overset{(c)}{=}& \frac{T_l \sigma_{\tau_{H_l}}^2(\mathbf{x})}{\lambda} \\
\overset{(d)}{=}& \frac{T_l \Sigma_{H_l}^2(\mathbf{x})}{\lambda} = \frac{T_l \Sigma_{H_l}^2(\mathbf{x})}{\sigma^2} \leqslant 2C^2 \gamma_{T_l},
\end{aligned}
\tag{90}
$$

where (a) is from $\Phi_{H_l}^\top \mathbf{W}_{H_l}^2 \Phi_{H_l} \leqslant \Phi_{H_l}^\top (T_l \mathbf{I}) \mathbf{W}_{H_l} \Phi_{H_l} = T_l \Phi_{H_l}^\top \mathbf{W}_{H_l} \Phi_{H_l}$ because each diagonal entry of W_{H_l} satisfies $[W_{H_l}]_{hh} = T_l(\mathbf{a}_h) \leqslant T_l$, (b) is based on Eq. (28), (c) is from Eq. (23), and (d) is according to the equivalence representation in Lemma B.1. The last step is from the result in Lemma B.3.

Substituting the above result into Eq. (89), we have

$$
\sigma_{\text{sum}}^2 \leqslant 2C^2 \gamma_{T_l} \sigma_{nc}^2 = \sigma_n^2. \tag{91}
$$

According to the tail bound of Gaussian variables, we have

$$
\mathbb{P}\left[\left| \mathbf{k}(\mathbf{x}, \mathbf{A}_{H_l})^\top (\mathbf{K}_{\mathbf{A}_{H_l}\mathbf{A}_{H_l}} + \lambda \mathbf{W}_{H_l}^{-1})^{-1} \boldsymbol{\rho} \right| \geqslant \sqrt{2\sigma_n^2 \log(1/\beta)} \right] \leqslant 2 \exp\left\{ -\frac{4C^2 \gamma_T \sigma_{nc}^2 \log(1/\beta)}{2\sigma_{\text{sum}}^2} \right\} \leqslant 2\beta.
$$

□

Proof of Theorem 6.3. Similar to the proof of Theorem 5.1, we, to prove Theorem 6.3, first present three results when the concentration inequality in Theorem E.7 holds, then obtain an upper bound for the regret incurred in a particular phase $l > 2$ with high probability, and finally sum up the regret over all phases.

1) Three observations when Eq. (84) holds

Define a "good" event when Eq. (84) holds in the l-th phase as:

$$
\tilde{\mathcal{E}}_l \triangleq \{ \forall \mathbf{x} \in \mathcal{D}_l, |f(\mathbf{x}) - \tilde{\mu}_l(\mathbf{x})| \leqslant \tilde{w}_l(\mathbf{x}) \}.
$$

We have $\mathbb{P}[\tilde{\mathcal{E}}_l] \geqslant 1 - 6|\mathcal{D}|\beta$ via the union bound. Then, similar to the non-private case, under event $\tilde{\mathcal{E}}_l$ in the l-th phase, we have the following three observations:

1. For any optimal action $\mathbf{x}^* \in \operatorname{argmax}_{\mathbf{x} \in \mathcal{D}} f(\mathbf{x})$, if $\mathbf{x}^* \in \mathcal{D}_l$, then $\mathbf{x}^* \in \mathcal{D}_{l+1}$.
2. Let $f^* = \max_{\mathbf{x} \in \mathcal{D}} f(\mathbf{x})$. Supposed that $x^* \in \mathcal{D}_l$. For any $\mathbf{x} \in \mathcal{D}_{l+1}$, its reward gap from the optimal reward is bounded by $4 \max_{\mathbf{x} \in \mathcal{D}_l} \tilde{w}_l(\mathbf{x})$, i.e.,

$$
f^* - f(\mathbf{x}) \leqslant 4 \max_{\mathbf{x} \in \mathcal{D}_l} \tilde{w}_l(\mathbf{x}).
$$

3. The confidence width function in the private setting satisfies

$$\max_{\mathbf{x}\in\mathcal{D}_l} \tilde{w}_l(\mathbf{x}) \leqslant \max_{\mathbf{x}\in\mathcal{D}_l} w_l(\mathbf{x}) + \frac{G_1\gamma_T\sqrt{2\log(1/\beta)}}{|U_l|}, \tag{92}$$

where $G_1 \triangleq \frac{8C^2\sqrt{2(\kappa^2+\sigma^2)\sigma^2\log(1/\delta_1)\ln(1.25/\delta_2)}}{\varepsilon\sqrt{C^2-1}}$.

The first two observations can be derived similar to the non-private case. Regarding the third observation, we have the confidence width function in the private setting $\tilde{w}_l(\mathbf{x}) = w_l(\mathbf{x}) + \sqrt{2\sigma_n^2\log(1/\beta)}$ and

$$\begin{aligned}
&\sqrt{2\sigma_n^2\log(1/\beta)}\\
&= 2C\sqrt{\gamma_T\sigma_{nc}^2\log(1/\beta)}\\
&= \frac{4C\sqrt{2(\kappa^2+\sigma^2)H_l\gamma_T\log(1/\delta_1)\ln(1.25/\delta_2)\log(1/\beta)}}{\varepsilon|U_l|}\\
&\overset{(a)}{\leqslant} \frac{8C^2\gamma_T\sqrt{2(\kappa^2+\sigma^2)\sigma^2\log(1/\delta_1)\ln(1.25/\delta_2)\log(1/\beta)}}{\varepsilon|U_l|\sqrt{C^2-1}}\\
&\leqslant \underbrace{\frac{8C^2\sqrt{2(\kappa^2+\sigma^2)\sigma^2\log(1/\delta_1)\ln(1.25/\delta_2)}}{\varepsilon\sqrt{C^2-1}}}_{G_1}\cdot\frac{\gamma_T\sqrt{2\log(1/\beta)}}{|U_l|},
\end{aligned}$$

where (a) is from Lemma 5.6.

2) Regret in a specific phase $l > 2$.
Under event $\tilde{\mathcal{E}}_{l-1}$, the regret incurred in the l-th phase is

$$\begin{aligned}
&\sum_{t\in\mathcal{T}_l} f^* - f(\mathbf{x}_t)\\
\leqslant&\sum_{t\in\mathcal{T}_l} 4\max_{\mathbf{x}\in\mathcal{D}_{l-1}}\tilde{w}_{l-1}(\mathbf{x})\\
\leqslant&4T_l\max_{\mathbf{x}\in\mathcal{D}_{l-1}} w_{l-1}(\mathbf{x})\\
\overset{(a)}{\leqslant}&4T_l\max_{\mathbf{x}\in\mathcal{D}_{l-1}} w_{l-1}(\mathbf{x}) + 4T_l\cdot\frac{G_1\gamma_T\sqrt{2\log(1/\beta)}}{|U_{l-1}|}\\
\leqslant&4T_l\max_{\mathbf{x}\in\mathcal{D}_{l-1}} w_{l-1}(\mathbf{x}) + 4G_1\gamma_T\sqrt{2\log(1/\beta)}2^{(1-\alpha)(l-1)}\\
\leqslant&4\sqrt{2\kappa^2\log(1/\beta)}\sqrt{2^{(2-\alpha)(l-1)}} + 8\sigma C\sqrt{2\gamma_T\log(1/\beta)}\sqrt{2^{(1-\alpha)(l-1)}} + 8\sigma BC\sqrt{\gamma_T 2^{l-1}}\\
&+ 4G_1\gamma_T\sqrt{2\log(1/\beta)}2^{(1-\alpha)(l-1)},
\end{aligned}$$

where (a) is from Observation 3 and the last step is from Eq. (59).

3) Total regret.
Define $\tilde{\mathcal{E}}_g$ as the event where the "good" event occurs in every phase in the private setting, i.e., $\tilde{\mathcal{E}}_g \triangleq \bigcap_{l=1}^{L}\tilde{\mathcal{E}}_l$. It is not difficult to obtain $\mathbb{P}[\mathcal{E}_g] \geqslant 1 - 6|\mathcal{D}|\beta L$ by applying union bound. At the

same time, the total regret under event $\tilde{\mathcal{E}}_g$ becomes

$$\begin{aligned}
R_g &= \sum_{l=1}^{L}\sum_{t\in\mathcal{T}_l}(f^* - f(\mathbf{x}_t)) \\
&\leqslant 2B\kappa + \sum_{l=2}^{L} 4\sqrt{2\kappa^2\log(1/\beta)}\sqrt{2^{(2-\alpha)(l-1)}} \\
&+ \sum_{l=2}^{L} 8\sigma C\sqrt{2\gamma_T\log(1/\beta)}\sqrt{2^{(1-\alpha)(l-1)}} \\
&+ \sum_{l=2}^{L} 8\sigma BC\sqrt{\gamma_T 2^{l-1}} + \sum_{l=2}^{L} 4G_1\gamma_T\sqrt{2\log(1/\beta)}2^{(1-\alpha)(l-1)} \\
&\leqslant 2B\kappa + 4\sqrt{2\kappa^2\log(1/\beta)}\cdot 4\sqrt{2^{(L-1)(2-\alpha)}} \\
&+ 8\sigma C\sqrt{2\gamma_T\log(1/\beta)}\cdot C_1\sqrt{2^{(1-\alpha)(L-1)}} \qquad \left(C_1 \triangleq \frac{\sqrt{2^{1-\alpha}}}{\sqrt{2^{1-\alpha}}-1}\right) \\
&+ 8\sigma BC\sqrt{\gamma_T}\cdot 4\sqrt{2^{L-1}} \\
&+ 4G_1\gamma_T\sqrt{2\log(1/\beta)}\cdot C_2 2^{(1-\alpha)(L-1)} \qquad \left(C_2 \triangleq \frac{2^{1-\alpha}}{2^{1-\alpha}-1}\right) \\
&\leqslant 2B\kappa + 16\sqrt{2\kappa^2\log(1/\beta)}T^{1-\alpha/2} + 8\sigma C_1 C\sqrt{2\gamma_T\log(1/\beta)T^{1-\alpha}} \\
&+ 32\sigma BC\sqrt{\gamma_T T} + 4C_2G_1\gamma_T\sqrt{2\log(1/\beta)}T^{1-\alpha},
\end{aligned} \tag{93}$$

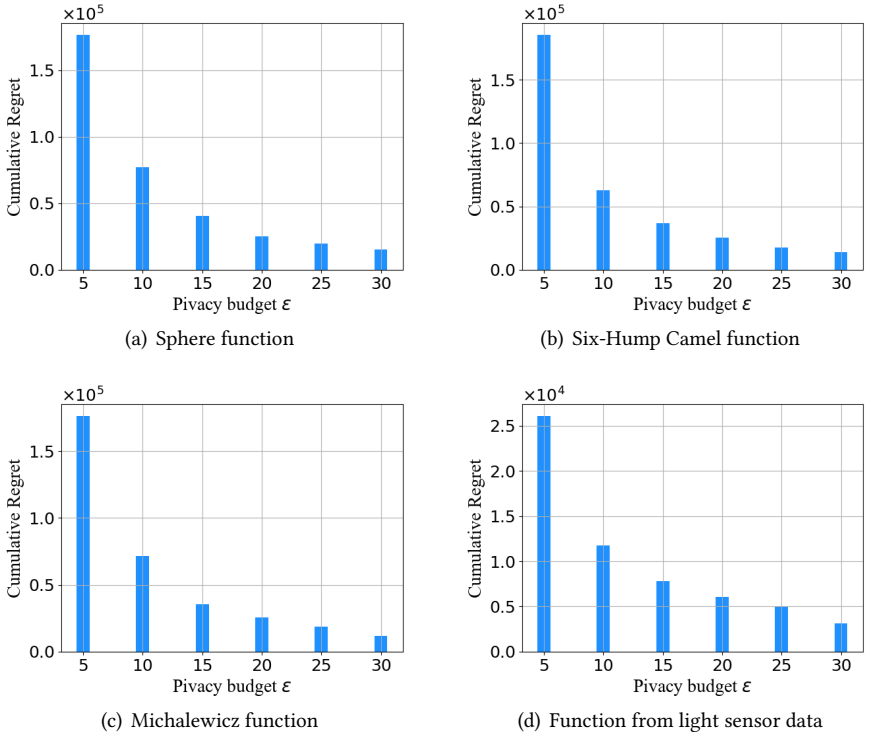
where the last step is is due to $2^{L-1} \leqslant T$ and $L \leqslant \log(2T)$ since $\sum_{l=1}^{L-1} T_l + 1 \leqslant T$. On the other hand, $R_b \leqslant 2B\kappa T$ since $|\max_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x}) - f(\mathbf{x})| \leqslant 2B\kappa$ for all $\mathbf{x} \in \mathcal{D}$. Choose $\beta = 1/(|\mathcal{D}|T)$ in Algorithm 2. Then, the expected regret is:

$$\begin{aligned}
\mathbb{E}[R(T)] &= \mathbb{P}[\tilde{\mathcal{E}}_g]R_g + (1-\mathbb{P}[\tilde{\mathcal{E}}_g])R_b \\
&\leqslant R_g + 6|\mathcal{D}|\beta L\cdot 2B\kappa T \\
&\leqslant 2B\kappa + 16\sqrt{2\kappa^2\log(1/\beta)}T^{1-\alpha/2} + 8\sigma C_1 C\sqrt{2\gamma_T\log(1/\beta)T^{1-\alpha}} + 32\sigma BC\sqrt{\gamma_T T} \\
&+ 4C_2G_1\gamma_T\sqrt{2\log(1/\beta)}T^{1-\alpha} + 12B\kappa|\mathcal{D}|\beta LT \\
&= 2B\kappa + 16T^{1-\alpha/2}\sqrt{2\kappa^2\log(|\mathcal{D}|T)} + 8\sigma C_1 C\sqrt{2\gamma_T T^{1-\alpha}\log(|\mathcal{D}|T)} + 32\sigma BC\sqrt{\gamma_T T} \\
&+ 4C_2G_1\gamma_T\sqrt{2\log(|\mathcal{D}|T)}T^{1-\alpha} + 12B\kappa\log(2T) \\
&= O(T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)}) + O(\sqrt{\gamma_T T^{1-\alpha}\log(|\mathcal{D})T)} + O(G_1\gamma_T T^{1-\alpha}\sqrt{\log(|\mathcal{D}|T)}) + O(\sqrt{\gamma_T T}).
\end{aligned} \tag{94}$$

Finally, substituting G_1 with $\delta_1 = \delta_2 = \delta/2$, we have the total expected regret under the DP-DPBE with the central model is

$$\begin{aligned}
\mathbb{E}[R(T)] = O(T^{1-\alpha/2}\sqrt{\log(|\mathcal{D}|T)}) + O\left(\frac{\ln(1/\delta)\gamma_T T^{1-\alpha}\sqrt{\log(kT)}}{\varepsilon}\right) \\
+ O(\sqrt{\gamma_T T^{1-\alpha}\log(|\mathcal{D})T)} + O(\sqrt{\gamma_T T}).
\end{aligned} \tag{95}$$

□



(a) Sphere function (b) Six-Hump Camel function

(c) Michalewicz function (d) Function from light sensor data

Fig. 8. Performance of DP-DPBE: Final cumulative regret vs. privacy budget ε with $\delta = 10^{-6}$.

While the DPBE algorithm uses GP tools to define and manage the uncertainty in estimating the unknown function f, the analysis of DPBE algorithm does not rely on any *Bayesian* assumption about f being actually drawn from the prior $\mathcal{GP}(0, k)$, and it only requires f to be bounded in the kernel norm associated with the RKHS $\mathcal{H}_k$.

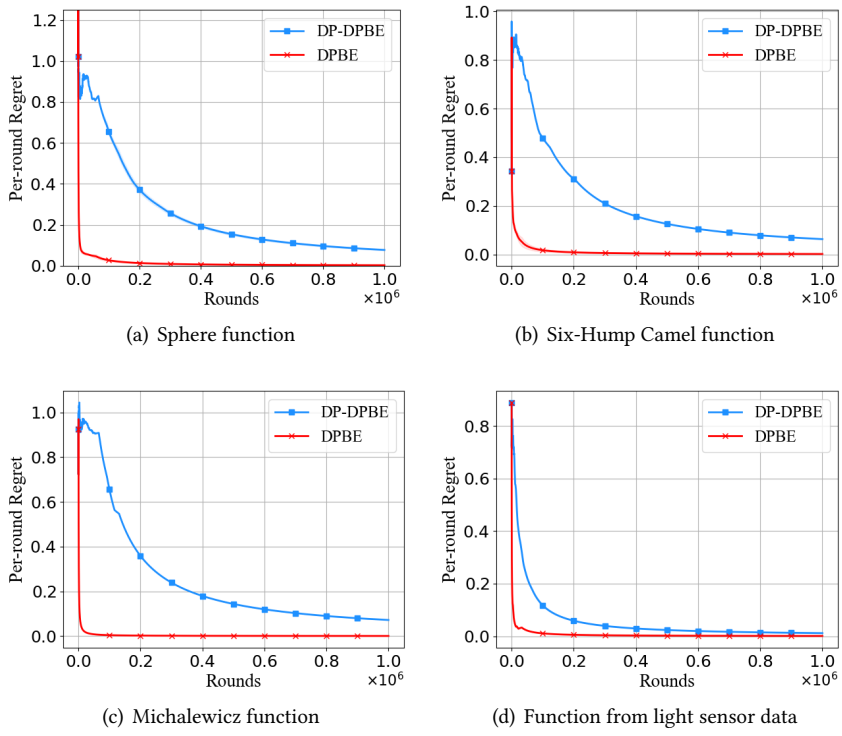
F ADDITIONAL NUMERICAL RESULTS

F.1 Evaluation of DP-DPBE

In Section 7, we evaluated DP-DPBE on the synthetic function. In this subsection, we present additional numerical results for DP-DPBE on the standard benchmark functions and the function from real-world (light-sensor) data. By considering the same setting as for the synthetic function, we run $T = 10^6$ rounds and present how the cumulative regret at the end of T varies with privacy budget $\varepsilon \in \{5, 10, 15, 20, 25, 30\}$ and $\delta = 10^{-6}$ in Figure 8. Then, by choosing privacy parameters $\delta = 10^{-6}$ and $\varepsilon = 15$, we also compare the per-round regret of DP-DPBE and DPBE for the three benchmark functions and the real-world (light-sensor) data and present the results in Figure 9. We perform 20 runs for each simulation. From these results, we make similar observations to those for the synthetic function: the privacy-regret tradeoff and achieving privacy "for free".

F.2 Comparison with State-of-the-Art

In Section 8, we provide simulation results on the regret performance and running time of GP-UCB, BPE, and our algorithm DPBE with different values of α on the synthetic data generated in Section 7.1



(a) Sphere function (b) Six-Hump Camel function

(c) Michalewicz function (d) Function from light sensor data

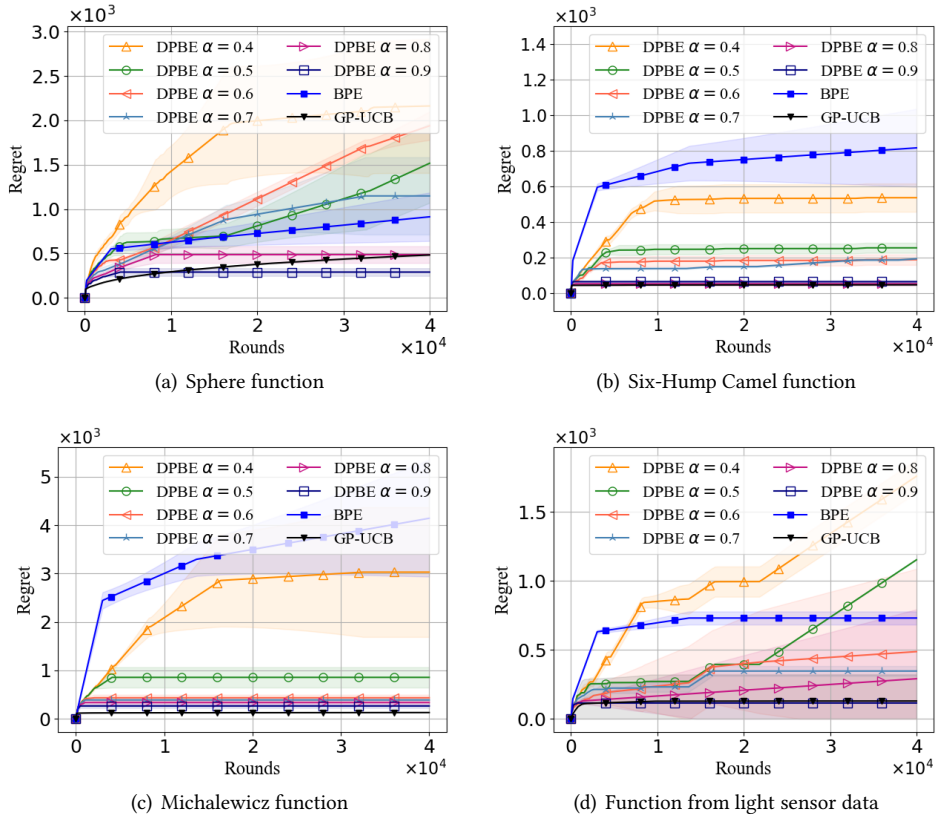
Fig. 9. Performance of DP-DPBE: Per-round regret vs. time with parameters $\varepsilon = 15$ and $\delta = 10^{-6}$.

Table 7. Comparison of running time (seconds) under GP-UCB, BPE, and DPBE with different values of α.

Algorithms	DPBE						GP-UCB	BPE
	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 0.8$	$\alpha = 0.9$		
Sphere	0.08	0.07	0.07	0.07	0.09	0.13	4.68	37.87
Six-Hump Camel	0.04	0.03	0.04	0.03	0.04	0.04	4.79	10.43
Michalewicz	0.04	0.04	0.05	0.06	0.07	0.11	4.95	4.48
Light Sensor Data	0.04	0.06	0.07	0.03	0.06	0.05	3.22	82.08

In this section, we add additional numerical results on three benchmark functions (Sphere, Six-hump Camel, Michalewicz) and one function from real-world data– Light sensor data [34]. The parameters of the problem setting and the algorithms are as follows: $T = 4 \times 10^4$, $|\mathcal{D}| = 100$, and $k = k_{SE}$ with $l_{SE} = 0.2$; (a) Sphere function. Settings: $d = 3, C = 1.5, \sigma = 0.01, v^2 = 0.001, \lambda = \sigma^2/v^2$; (b) Six-Hump Camel function. Settings: $d = 2, C = 1.5, \sigma = 0.01, v^2 = 0.01, \lambda = \sigma^2/v^2$; (c) Michalewicz function. Settings: $d = 2, C = 1.5, \sigma = 0.01, v^2 = 0.01, \lambda = \sigma^2/v^2$; (d) Functions from real-world data. Settings: $d = 2, C = 1.42, \sigma = 0.01, v^2 = 0.01, \lambda = \sigma^2/v^2$. We plot the cumulative regret for all the algorithms in Figure 10 and present the running time in Table 7.

Received August 2022; revised October 2022; accepted January 2023



(a) Sphere function

(b) Six-Hump Camel function

(c) Michalewicz function

(d) Function from light sensor data

Fig. 10. Comparison of regret performance under DPBE, GP-UCB, and BPE on three benchmark functions and one function from real-world dataset. The shaded area represents the standard deviation.