Order-Optimal Regret in Distributed Kernel Bandits using Uniform Sampling with Shared Randomness

Nikola Pavlovic
Cornell University

Sudeep Salgia
Carnegie Mellon

Qing Zhao
Cornell Univesrity

Abstract

We consider distributed kernel bandits where N agents aim to collaboratively maximize an unknown reward function that lies in a reproducing kernel Hilbert space. Each agent sequentially queries the function to obtain noisy observations at the query points. Agents can share information through a central server, with the objective of minimizing regret that is accumulating over time T and aggregating over agents. We develop the first algorithm that achieves the optimal regret order (as defined by centralized learning) with a communication cost that is sublinear in both N and T. The key features of the proposed algorithm are the uniform exploration at the local agents and shared randomness with the central server. Working together with the sparse approximation of the GP model, these two key components make it possible to preserve the learning rate of the centralized setting at a diminishing rate of communication.

1 INTRODUCTION

1.1 Distributed Kernel Bandits

We study the problem of zeroth-order online stochastic optimization in a distributed setting, where N agents aim to collaboratively maximize a reward function with communications facilitated by a central server. The reward function $f : \mathcal{X} \to \mathbb{R}$ is unknown; it is only known that it lives in a Reproducing Kernel Hilbert Space (RKHS) associated with a known kernel k. Each agent sequentially chooses points in the function domain $\mathcal{X}$ to query and subsequently receives noisy feedback on the function values (i.e., random rewards) at the query points. The goal is for each distributed agent to converge quickly to $x^* \in \arg\max_{x\in\mathcal{X}} f(x)$, a global maximizer of f. We quantify this goal as minimizing the cumulative regret summed over a learning horizon of length T and over all N agents:

$$R = \sum_{n=1}^{N}\sum_{t=1}^{T} \left(f(x^*) - f(x_t^{(n)})\right), \qquad (1)$$

where $x_t^{(n)}$ denotes the point queried by agent n at time t.

The above zeroth-order stochastic optimization problem can be viewed as a continuum-armed kernelized-bandit problem [Srinivas et al., 2010]. The expressive power of the RKHS model represents a broad family of objective functions. In particular, it is known that the RKHS of typical kernels, such as the Matérn family of kernels, can approximate almost all continuous functions on compact subsets of $\mathbb{R}^d$ [Srinivas et al., 2010]. The problem has been studied extensively under a centralized setting with a single decision maker (i.e., $N = 1$), for which several algorithms have been proposed, including UCB-based algorithms [Srinivas et al., 2010, Chowdhury and Gopalan, 2017, Abbasi-Yadkori et al., 2011], batched pure exploration [Li and Scarlett, 2022], tree-based domain shrinking [Salgia et al., 2021] and RIPS [Camilleri et al., 2021]. Optimal learning efficiency in terms of regret order in T has been obtained in both the stochastic [Li and Scarlett, 2022, Salgia et al., 2021] and the contextual setting [Valko et al., 2013].

In addition to learning efficiency, distributed kernel bandits face a new challenge of communication efficiency. Without constraints on the communication overhead, all agents can share their local observations and coordinate their individual query actions at no cost. The distributed problem can be trivially reduced to a centralized one. At the other end of the spectrum is a complete decoupling of the agents, resulting in N independent single-user problems without the benefit of data sharing for accelerated learning. The tension between learning efficiency (which demands data sharing

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258. Copyright 2025 by the author(s).

and action coordination) and communication efficiency is evident. A central question to this trade-off is how to achieve the optimal learning rate enjoyed by the centralized setting using a minimum amount of message exchange among agents.

In contrast to the extensive literature on centralized kernel bandits, distributed kernel bandits are much less explored despite their broad applications (e.g., federated learning for hyperparameter tuning [Dai et al., 2020] and collaborative training of neural nets using the recent theory of Neural Tangent Kernel [Jacot et al., 2018]). There exist only a handful of studies under drastically different settings and constraints (see Sec.1.3). For the setting considered in this work, no distributed learning algorithms exist that achieve the optimal regret order with a sublinear (in both T and N) message exchange among agents.

1.2 Main Results

In this paper, we develop the first algorithm for distributed kernel bandits that achieves the optimal order of regret enjoyed by centralized learning with a sublinear message exchange in both T and N.

To tackle the essential tradeoff between learning rate and communication efficiency, a distributed learning algorithm needs a communication strategy that governs *what* to communicate and *how to integrate* the shared information into local query actions. To minimize the total regret that is accumulating over time and aggregating over the agents, the communication strategy needs to work in tandem with the query actions to ensure a continual flow of information available at all agents for decision-making.

A natural answer to *what* to communicate in a distributed learning problem is certain sufficient local statistics of the underlying unknown parameters. For example, for multi-armed (i.e., discrete arms) and linear bandits, this corresponds to the local estimates of the arm mean values and the mean reward vector respectively. However, for kernel bandits, the corresponding quantity would be an estimate of the function, which is potentially infinite-dimensional and hence an impractical choice for communication. Existing studies resolve this issue by exchanging local query actions and observations across all agents and throughout the learning horizon [Li et al., 2022, Dubey and Pentland, 2020], resulting in a communication cost growing linearly in both N and T.

Even with a communication cost growing linearly in both N and T, preserving the full learning power of a centralized decision maker with NT query points is not immediate. The prevailing approaches to centralized kernel bandits that achieve order optimal regret build on the maximum posterior variance (MPV) sampling strategy [Li and Scarlett, 2022] which queries, at each time, the point with the highest posterior variance conditioned on all past observations. Ensuring such a maximal uncertainty reduction at each query point is believed to be crucial in utilizing the full statistical power of all query points. Unfortunately, such a fully adaptive query strategy is incompatible with the parallel learning among distributed agents. To emulate the MPV sampling at each of the NT query points would require the agents to *take turns* in their queries and share the local observations immediately with all other agents, an infeasible strategy for most distributed learning problems. Implementing MPV-based sampling in parallel across agents, however, loses the full adaptivity. This is arguably the main obstacle in realizing the optimal learning rate of a centralized kernel bandit in a distributed setting.

To tackle the above challenges, our proposed algorithm represents major departures from the prevailing approaches. Referred to as DUETS (Distributed Uniform Exploration of Trimmed Sets), this algorithm has two key features: *uniform exploration* at the local agents and *shared randomness* with the central server.

In DUETS , each agent employs uniform (at random) sampling as the query strategy. Uniform sampling is fully compatible with parallel learning. In particular, note that the union of the local sets of size t query points obtained at the agents through uniform sampling is identical (in distribution) to the set of size Nt query points obtained at a centralized decision maker using the same uniform sampling strategy. This superposition property of uniform sampling allows us to leverage the recent results on random exploration in centralized kernel bandits [Salgia et al., 2023a], and is crucial in achieving the optimal learning rate defined by the centralized setting. In addition to preserving the learning rate of the centralized setting, uniform sampling enjoys advantages in computation as well as communication aspects. Comparing with the MPV strategy that requires an expensive maximization of a non-convex acquisition function for finding each query point, uniform sampling is extremely simple to implement. This computational efficiency can be particularly attractive to distributed local devices. In terms of communication efficiency, uniform sampling makes it possible to bypass the exchange of query points altogether and reduce the exchange of reward observations through the *shared randomness* strategy detailed below.

In DUETS, each agent has access to an independent coin, i.e., a source of randomness, which is unknown to the other agents but is known to the server. The shared randomness enables the server to reproduce the points queried by the agents, thereby resulting in effective transmission of the local set of queried points at each agent to the server at *no communication cost* (Please refer to Sec. 3 for an additional discussion).

To reduce the communication overhead associated with the reward observations, we employ sparse approximation of GP models [Wild et al., 2021]. The availability of *all* the queried points at the server provides the perfect platform for leveraging the power of sparse approximation to reduce the communication to a diminishing fraction of the total number of observations. Specifically, the server, with access to all the query points, selects a small subset of points that can approximate, to sufficient accuracy, the posterior statistics corresponding to all the points queried by the agents. This allows a diminishing rate of communication to share local reward observations. It is this integration of uniform sampling, shared randomness, and sparse approximation in DUETS that makes it possible to achieve the optimal learning rate of the centralized setting at a communication cost that is sublinear in both N and T.

We analyze the performance of DUETS and establish that it incurs a cumulative regret of $\widetilde{\mathcal{O}}(\sqrt{NT\gamma_{NT}}\log(T/\delta))$[1] with probability $1-\delta$, where γ_{NT} denotes the maximal information gain of the kernel and represents the effective dimension of the kernel. Note that this matches the lower bound (up to logarithmic factors) for any centralized algorithm with a total of NT queries as established in Scarlett et al. [2017], thereby establishing the order-optimality of the proposed algorithm. To the best knowledge of the authors, this is the *first* algorithm to achieve the optimal order of regret for the problem of distributed kernel bandits. We also establish a bound of $\widetilde{\mathcal{O}}(\gamma_{NT})$ on the communication cost incurred by DUETS , where communication cost is measured by the number of real numbers transmitted during the algorithm (See Section 2 for more details). This significantly improves over the state-of-the-art of $\mathcal{O}(N\gamma_{NT}^3)$ achieved by ApproxDisKernelUCB proposed by Li et al. [2022] and is guaranteed to be sublinear in the total number of queries, NT.

1.3 Related Work

The existing literature on distributed kernel bandits is relatively slim. The most relevant to our work is that by Li et al. [2022], where the authors consider the problem of distributed contextual kernel bandits and propose a UCB based policy with sparse approximation of GP models and intermittent communication. Their proposed policy was shown to incur a cumulative regret of $\widetilde{\mathcal{O}}(\sqrt{NT}\gamma_{NT})$ and communication cost of $\mathcal{O}(N\gamma_{NT}^3)$. The DUETS algorithm proposed in this work, offers an improvement over the algorithm in Li et al. [2022] both in terms of regret and communication cost. While the contextual setting with varying arm action sets considered in their work is more general that the setting with a fixed arm set considered in this work, their proposed algorithm does not offer non-trivial reduction in regret or communication cost in the fixed arm setting. Moreover, both the regret and communication cost incurred by the algorithm in Li et al. [2022] are not guaranteed to be sublinear in the total number of queries, NT, for all kernels. Consequently, their algorithm does not guarantee convergence to x^* or a non-trivial communication cost for all kernels. On the other hand, both regret and communication cost of DUETS is guaranteed to be sub-linear implying both convergence and communication efficiency.

Among other studies, Du et al. [2023] consider the problem of distributed pure exploration in kernel bandits over finite action set, where they focus on designing learning strategies with low simple regret. In this work, we consider the more challenging continuum-armed setup with a focus on minimizing cumulative regret as opposed to simple regret. Another line of work explores impact of heterogeneity among clients and design algorithms to minimize this impact. Salgia et al. [2023b] consider personalized kernel bandits in which agents have heterogeneous models and aim to optimize the weighted sum of their own reward function and the average reward function over all the agents. Dubey and Pentland [2020] consider heterogeneous distributed kernel bandits over a graph in which they use additional kernel-based modeling to measure task similarity across different agents.

In contrast to the distributed kernel bandit, the problems of distributed multi-armed bandits and linear bandits have been extensively studied. For distributed multi-armed bandits (MAB), a variety of algorithms have been proposed for distributed learning under different network topologies [Landgren et al., 2017, Shahrampour et al., 2017, Sankararaman et al., 2019, Chawla et al., 2020, Zhu et al., 2021]. Shi et al. [2021] and Shi and Shen [2021] have analyzed the impact of heterogeneity among agents in the distributed MAB problem. Similarly, the problem of distributed linear bandits is also well-understood in variety of settings with different network topologies [Korda et al., 2016], heterogeneity among agents [Mitra et al., 2021, Ghosh et al., 2021, Hanna et al., 2022] and communication constraints [Mitra et al., 2022, Wang et al., 2019, Huang et al., 2021, Amani et al., 2022, Salgia and Zhao, 2023].

[1]The notation $\tilde{\mathcal{O}}(\cdot)$ hides poly-logarithmic factors.

2 PROBLEM FORMULATION

We consider a distributed learning framework consisting of N agents indexed by $\{1, 2, \ldots, N\}$. Under this framework, we study the problem of collaboratively maximizing an unknown function $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X} \subset \mathbb{R}^d$ is a compact, convex set. The function f belongs to the Reproducing Kernel Hilbert Space (RKHS), $\mathcal{H}_k$, associated with a known positive definite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The RKHS, $\mathcal{H}_k$, is a Hilbert space that is endowed by with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$ that obeys the reproducing property, i.e., $\langle g, k(x, \cdot)\rangle_{\mathcal{H}_k} = g(x)$ $\forall\, g \in \mathcal{H}_k$, and induces the norm $\|g\|_{\mathcal{H}_k} = \langle g, g\rangle_{\mathcal{H}_k}$.

The agents can access the unknown function by querying the function at different points in the domain $\mathcal{X}$. Upon querying a point $x \in \mathcal{X}$, the agent receives a reward $y = f(x) + \epsilon$, where ϵ is a noise term. We make the following assumptions on the unknown function f and noise.

Assumption 2.1. The RKHS norm of the function f is bounded by a known constant B, i.e., $\|f\|_{\mathcal{H}_k} \leq B$.

Assumption 2.2. The noise term ϵ is assumed to be independent across all agents and all queries and is a zero-mean, R sub-Gaussian random variable i.e., it satisfies the relation $\mathbb{E}[\exp(\lambda\epsilon)] \leq \exp\frac{\lambda^2 R^2}{2}$ for all $\lambda \in \mathbb{R}$.

Assumption 2.3. For each $r \in \mathbb{N}$, there exists a discretization $\mathcal{U}_r$ of $\mathcal{X}$ with $|\mathcal{U}_r| = \text{poly}(r)^2$ such that, for any $f \in \mathcal{H}_k$, we have $|f(x) - f([x]_{\mathcal{U}_r})| \leq \frac{\|f\|_{\mathcal{H}_k}}{r}$, where $[x]_{\mathcal{U}_r} = \arg\min_{x' \in \mathcal{U}_r} \|x - x'\|_2$.

Assumption 2.4. Let $\mathcal{L}_\eta = \{x \in \mathcal{X} | f(x) \geq \eta\}$ denote the level set of f for $\eta \in [-B, B]$. We assume that for all $\eta \in [-B, B]$, $\mathcal{L}_\eta$ is a disjoint union of at most $M_f < \infty$ components, each of which is closed and connected. Moreover, for each such component, there exists a bi-Lipschitzian map between each such component and $\mathcal{X}$ with normalized Lipschitz constant pair $L_f, L'_f < \infty$.

Assumptions 2.1-2.3 are standard, mild assumptions that are commonly adopted in the literature [Srinivas et al., 2010, Chowdhury and Gopalan, 2017, Li and Scarlett, 2022, Vakili et al., 2022, 2021a]. The existence of the discretization $\mathcal{U}_r$ in Assumption 2.3 has been justified and adopted in previous studies [Srinivas et al., 2010, Vakili et al., 2021a]. In particular, the popular class of kernels like Squared Exponential and Matérn kernels are known to be Lipschitz continuous, in which case a ε-cover of the domain with $\varepsilon = \mathcal{O}(1/r)$ is sufficient to show the existence of such a discretization. At a high level, Assumption 2.4 ensures that the structure of the levels sets of f satisfy a mild regularity condition. This is a mild assumption on f that we require to adopt a result from Salgia et al. [2023a] for our analysis. We also note that our assumption could be replaced by a weaker one, where we require the number of connected components be finite only for $\eta \geq c \max_{x\in\mathcal{X}} f(x)$, where c is an arbitrary constant $c \in (0, 1)$. This weaker assumption is provably satisfied[Johnson, 2015] by the "hard-instance" function used in proving the lower bound in Scarlett et al. [2017]. Thus adopting assumption 2.4 does not increase the lower-bound found in Scarlett et al. [2017].

The agents collaborate with each other by communicating through a central server. At each time instant, each agent can send a message to the server through the uplink channel. Based on the messages from different agents received by the server, it can then broadcast a message back to all the agents through the downlink channel.

Our objective is to design a distributed learning policy π that specifies for each agent n, the point $x_t^{(n)}$ to be queried at each time instant t, based on the information available at that agent up to time instant t. The performance of a collaborative learning policy π is measured through its performance in terms of both learning and communication efficiency over a learning horizon of T steps. The learning efficiency is measured using the notion of cumulative regret, as defined in (1).

The communication efficiency is measured using the sum of the uplink and downlink communication costs. In particular, let $C_{\text{up}}^{(n)}(T)$ denote the number of real numbers sent by the agent n to the server over the time horizon. The uplink cost of π, $C_{\text{up}}^{\pi}(T)$ is then given as the average communication cost over all agents:

$$C_{\text{up}}^{\pi}(T) = \frac{1}{N}\sum_{n=1}^{N} C_{\text{up}}^{(n)}(T). \tag{2}$$

Similarly, the downlink cost of π, $C_{\text{down}}^{\pi}(T)$ is given as the number of real numbers broadcast by the server over the entire time horizon averaged over all agents . The overall communication cost of π, $C^{\pi}(T)$, is given as $C^{\pi}(T) = C_{\text{up}}^{\pi}(T) + C_{\text{down}}^{\pi}(T)$.

The objective is to design a distributed learning policy that achieves the order-optimal cumulative regret and incurs a low communication cost. We aim to provide high probability bounds on both the cumulative regret and communication cost that hold with probability $1 - \delta$ for any given $\delta \in (0, 1)$. We overview the basis of Gaussian Process models and their sparse approximation, both of which are central to our proposed policy.

[2]The notation $g(x) = \text{poly}(x)$ is equivalent to $g(x) = \mathcal{O}(x^k)$ for some $k \in \mathbb{N}$.

2.1 GP Models

In this section we present a brief overview of Gaussian Process models and their application on establishing confidence interval for RKHS elements.

A Gaussian Process (GP) is a random process G indexed by $\mathcal{X}$ and is associated with a mean function $\mu : \mathcal{X} \to \mathbb{R}$ and a positive definite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The random process G is defined such that for all finite subsets of $\mathcal{X}$, $\{x_1, x_2, \ldots, x_m\} \subset \mathcal{X}$, $m \in \mathbb{N}$, the random vector $[G(x_1), G(x_2), \ldots, G(x_m)]^\top$ follows a multivariate Gaussian distribution with mean vector $[\mu(x_1), \ldots, \mu(x_n)]]^\top$ and covariance matrix $\Sigma = [k(x_i, x_j)]_{i,j=1}^m$. Throughout the work, we consider GPs with $\mu \equiv 0$. When used as a prior for a data generating process under Gaussian noise, the conjugate property provides closed form expressions for the posterior mean and covariance of the GP model. Specifically, given a set of observations $\{\mathbf{X}_m, \mathbf{Y}_m\} = \{(x_i, y_i)\}_{i=1}^m$ from the underlying process, the expression for posterior mean and variance of GP model is given as follows:

$$\mu_m(x) = k_{\mathbf{X}_m}(x)^\top (\lambda \mathbf{I}_m + \mathbf{K}_{\mathbf{X}_m, \mathbf{X}_m})^{-1} \mathbf{Y}_m, \tag{3}$$

$$\sigma_m^2(x) = (k(x,x) - k_{\mathbf{X}_m}^\top(x)(\lambda \mathbf{I}_m + \mathbf{K}_{\mathbf{X}_m, \mathbf{X}_m})^{-1} k_{\mathbf{X}_m}(x)). \tag{4}$$

In the above expressions, $k_{\mathbf{X}_m}(x) = [k(x_1, x), k(x_2, x) \ldots k(x_n, x)]^\top$, $\mathbf{K}_{\mathbf{X}_m, \mathbf{X}_m} = \{k(x_i, x_j)\}_{i,j=1}^m$, $\mathbf{I}_m$ is the $m \times m$ identity matrix and λ is the variance of the Gaussian noise.

Following a standard approach in the literature [Srinivas et al., 2010], we model the data corresponding to observations from the unknown f, which belongs to the RKHS of a positive definite kernel k, using a GP with the same covariance kernel k. In particular, we assume a *fictitious* GP prior over the fixed, unknown function f along with *fictitious* Gaussian distribution for the noise. The benefit of this approach is that the posterior mean and variance of this GP model serve as tools to both predict the values of the function f and quantify the uncertainty of the prediction at unseen points in the domain, as shown by the following lemma .

Lemma 2.5. *Vakili et al. [2021a, Thm. 1] Assume that 2.1 and 2.2 hold. Given a set of observations $\{\mathbf{X}_m, \mathbf{Y}_m\}$ as described above, such that the query points $\mathbf{X}_m$ are chosen independent of the noise sequence, then for a fixed $x \in \mathcal{X}$, the following relation holds with probability at least $1 - \delta$:*

$$|h(x) - \mu_m(x)| \leq \beta(\delta) \cdot \sigma_m(x),$$

where $\beta(\delta) = B + R\sqrt{(2/\lambda)\log(2/\delta)}$.

We would like to emphasize that these assumptions are modeling techniques used as a part of algorithm and not a part of the problem setup. In particular, the function f is *fixed, deterministic* function in $\mathcal{H}_k$ and the noise is R-sub-Gaussian.

Lastly, given a set of points $\mathbf{X}_m = \{x_1, x_2, \ldots, x_m\} \in \mathcal{X}$, the information gain of the set $\mathbf{X}_m$ is defined as $\gamma_{\mathbf{X}_m} := \frac{1}{2}\log(\det(\mathbf{I}_m + \lambda^{-1}\mathbf{K}_{\mathbf{X}_m, \mathbf{X}_m}))$. Using this, we can define the maximal information gain of a kernel as $\gamma_m := \sup_{\mathbf{X}_m} \gamma_{\mathbf{X}_m}$. Maximal information gain is closely related to the effective dimension of a kernel [Calandriello et al., 2019] and helps characterize the regret performance of kernel bandit algorithms [Srinivas et al., 2010, Chowdhury and Gopalan, 2017]. γ_m depends only the kernel and λ and has been shown to be an increasing sublinear function of m [Srinivas et al., 2010, Vakili et al., 2021b].

2.2 Sparse approximation of GP models

The sparsification of GP models refers to the idea of approximating the posterior mean and variance of a GP model, corresponding to a set of observations $\{\mathbf{X}_m, \mathbf{Y}_m\}$, using a subset of query points $\mathbf{X}_m$. In particular, let $\mathcal{S}$ be a subset of $\mathbf{X}_m$ consisting of $r < m$ points. The approximate posterior mean and variance based on points in $\mathcal{S}$, referred to as the inducing set, is given as [Wild et al., 2021].

$$\tilde{\mu}_m(x) = z_{\mathcal{S}}(x)^\top \left(\lambda \mathbf{I}_{|\mathcal{S}|} + \mathbf{Z}_{\mathbf{X}_m, \mathcal{S}}^\top \mathbf{Z}_{\mathbf{X}_m, \mathcal{S}}\right)^{-1} \mathbf{Z}_{\mathcal{X}_m, \mathcal{S}}^\top \mathbf{Y}_m \tag{5}$$

$$\lambda \tilde{\sigma}_m^2(x) = k(x,x) - \tag{6}$$

$$- z_{\mathcal{S}}^\top(x) \mathbf{Z}_{\mathbf{X}_m, \mathcal{S}}^\top \mathbf{Z}_{\mathbf{X}_m, \mathcal{S}} \left(\lambda \mathbf{I}_{|\mathcal{S}|} + \mathbf{Z}_{\mathbf{X}_m, \mathcal{S}}^\top \mathbf{Z}_{\mathbf{X}_m, \mathcal{S}}\right)^{-1} z_{\mathcal{S}}(x)$$

where $z_{\mathcal{S}}(x) = \mathbf{K}_{\mathcal{S},\mathcal{S}}^{-\frac{1}{2}} k_{\mathcal{S}}(x)$ and $\mathbf{Z}_{\mathbf{X}_m, \mathcal{S}} = [z_{\mathcal{S}}(x_1), z_{\mathcal{S}}(x_2), \ldots, z_{\mathcal{S}}(x_m)]^\top$.

Note that it is sufficient to know the matrix $\mathbf{Z}_{\mathcal{X}_m, \mathcal{S}}^\top \mathbf{Z}_{\mathcal{X}_m, \mathcal{S}} \in \mathbb{R}^{r \times r}$, vector $\mathbf{Z}_{\mathcal{X}_m, \mathcal{S}}^\top \mathbf{Y}_m \in \mathbb{R}^r$ and the set $\mathcal{S}$ in order for $\tilde{\mu}$ and $\tilde{\sigma}$ to be calculated.

3 THE DUETS ALGORITHM

In this section, we present the proposed algorithm DUETS. We first describe the randomization at each agent and the shared randomness with the server. Each agent n has a coin $\mathscr{C}_n$ for generating *truly* random bits that are independent of those generated by other agents. Each agent's coin is unknown to other agents, but known to the central server. As a result, the server can reproduce the random bits generated at all agents.

DUETS employs an epoch-based elimination structure where the domain $\mathcal{X}$ is successively trimmed across epochs to maintain an active region that contains a global maximizer x^* with high probability for future exploration. Specifically, in each epoch j, the server and the agents maintain a common active subset of the domain $\mathcal{X}_j \subseteq \mathcal{X}$ with $\mathcal{X}_1$ initialized to $\mathcal{X}$. The operations in each epoch are as follows.

During the j^{th} epoch, each agent n, using its private coin $\mathscr{C}_n$, generates $\mathcal{D}_j^{(n)}$, a set of T_j points that are uniformly distributed in the set $\mathcal{X}_j$[3]. T_j is set to $\lfloor\sqrt{TT_{j-1}}\rfloor$, with T_1 being an input to the algorithm. Each agent n queries all the points in $\mathcal{D}_j^{(n)}$ and obtains $\mathbf{Y}_j^{(n)} \in \mathbb{R}^{T_j}$, the corresponding vector of reward observations.

Since the server has access to the coins of all the agents, it can faithfully reproduce the set $\mathcal{D}_j = \bigcup_{n=1}^N \mathcal{D}_j^{(n)}$ without any communication between the server and the agents. In order to efficiently communicate the observed reward values from the agents to the server, we leverage sparse approximation of GP models along with the knowledge of the set $\mathcal{D}_j$ at the server. The server constructs a global inducing set $\mathcal{S}_j$ by including each point in $\mathcal{D}_j$ with probability $p_j := p_0\sigma_{j,\max}^2$, independent of other points where $\sigma_{j,\max}^2 = \sup_{x\in\mathcal{X}_j}\sigma_j^2(x)$ and $\sigma_j^2(\cdot)$ is the posterior variance corresponding to points collected in $\mathcal{D}_j$. Here, $p_0 = 72\log\left(\frac{4NT}{\delta'}\right)$ is an appropriately chosen universal constant which ensures that the approximate posterior statistics constructed using $\mathcal{S}_j$ are a faithful representation of the true posterior statistics corresponding to the set $\mathcal{D}_j$ with probability $1-\delta$. The server broadcasts the inducing set $\mathcal{S}_j$ to all the agents.

Upon receiving the inducing set, each agent n computes the projection $v_j^{(n)} \in \mathbb{R}^{|\mathcal{S}_j|}$ of its reward vector onto the inducing set as follows:

$$v_j^{(n)} := \mathbf{Z}_{\mathcal{D}_j^{(n)},\mathcal{S}_j}^\top \mathbf{Y}_j^{(n)}. \tag{7}$$

Each agent then sends back the lower-dimensional projected observations $v_j^{(n)}$ to the server, which subsequently aggregates them to obtain the vector $\overline{v}_j$ given as

$$\overline{v}_j := \left(\lambda\mathbf{I}_{|\mathcal{S}_j|} + \mathbf{Z}_{\mathcal{D}_j,\mathcal{S}_j}^\top \mathbf{Z}_{\mathcal{D}_j,\mathcal{S}_j}\right)^{-1}\left(\sum_{n=1}^N v_j^{(n)}\right). \tag{8}$$

Note that the summation $\sum_{n=1}^N v_j^{(n)}$ equals to $\mathbf{Z}_{\mathcal{D}_j,\mathcal{S}_j}^\top \mathbf{Y}_j$, i.e., projection of the rewards of all agents onto the inducing set. The server then broadcasts the vector $\overline{v}_j$ and $\sigma_{j,\max}$ to all the agents. The benefit of sending $\overline{v}_j$ as opposed to the sum of rewards is that it allows the agents to compute the posterior mean at the agents using their knowledge of the inducing set $\mathcal{S}_j$ (See. Eqn (5)).As the last step of the epoch, all the agents and the server trim the current set $\mathcal{X}_j$ to $\mathcal{X}_{j+1}$ using the following update rule:

$$\mathcal{X}_{j+1} = \left\{x \in \mathcal{X}_j : \tilde{\mu}_j(x) \geq \sup_{x'\in\mathcal{X}_j}\tilde{\mu}_j(x') - 2\beta(\delta')\sigma_{j,\max}\right\}, \tag{9}$$

[3] If the active region consists of multiple disjoint regions, then we carry out this step for each region separately. For simplicity of description, we assume the active region consists of a single connected component.

Algorithm 1 DUETS : Agent $n \in \{1,2,\ldots,N\}$

```
1: **Input**: Size of the first epoch $T_1$, error probability $\delta$
2: $t \leftarrow 0, j \leftarrow 1, \mathcal{X}_1 \leftarrow \mathcal{X}$
3: **while** $t < T$ **do**
4:     $\mathcal{D}_j^{(n)} = \emptyset$
5:     **for** $i \in \{1,2,\ldots,T_j\}$ **do**
6:         Query a point $x_t^{(n)}$ uniformly at random from $\mathcal{X}_j$ using the coin $\mathscr{C}_n$ and observe $y_t^{(n)}$
7:         $\mathcal{D}_j^{(n)} \leftarrow \mathcal{D}_j^{(n)} \cup \{x_t^{(n)}\}$
8:         $t \leftarrow t+1$
9:         **if** $t > T$ **then** Terminate
10:    **end for**
11:    Receive the global inducing set $\mathcal{S}_j$
12:    Set $v_j^{(n)} \leftarrow \mathbf{Z}_{\mathcal{D}_j^{(n)},\mathcal{S}_j}^\top \mathbf{Y}_j^{(n)}$, where $\mathbf{Y}_j^{(n)} = [y_{t-T_j}, y_{t-T_j+1}, \ldots, y_t]^\top$
13:    Receive $\overline{v}_j$ and $\sigma_{j,\max}$ from the server and compute $\tilde{\mu}_j(\cdot) = z_{\mathcal{S}_j}^\top(\cdot)\overline{v}_j$
14:    Update $\mathcal{X}_j$ to $\mathcal{X}_{j+1}$ using Eqn. (9)
15:    $T_{j+1} \leftarrow \lfloor\sqrt{TT_j}\rfloor$
16:    $j \leftarrow j+1$
17: **end while**
```

Algorithm 2 DUETS : Server

```
1: **input**: Size of the first epoch $T_1$, error probability $\delta$
2: $t \leftarrow 0, j \leftarrow 1, \mathcal{X}_1 \leftarrow \mathcal{X}$
3: **while** $t < T$ **do**
4:     Use the coins $\mathscr{C}_1, \mathscr{C}_2, \ldots, \mathscr{C}_N$ to reproduce the sets $\mathcal{D}_j^{(1)}, \mathcal{D}_j^{(2)}, \ldots, \mathcal{D}_j^{(N)}$
5:     $\mathcal{D}_j \leftarrow \cup_{n=1}^N \mathcal{D}_j^{(n)}$
6:     Set $\sigma_{j,\max} \leftarrow \sup_{x\in\mathcal{X}_j}\sigma_j(x)$
7:     Construct the set $\mathcal{S}_j$ by independently including each point from $\mathcal{D}_j$ with probability $p_j$
8:     Broadcast $\mathcal{S}_j$ to all the agents and receive $v_j^{(n)}$ from all agents $n \in \{1,2,\ldots,N\}$
9:     Set $\overline{v}_j$ using Eqn. (8)
10:    Broadcast $\overline{v}_j$ and $\sigma_{j,\max}$ to all the agents
11:    Update $\mathcal{X}_j$ to $\mathcal{X}_{j+1}$ using Eqn. (9)
12:    $t \leftarrow t + T_j$, $T_{j+1} \leftarrow \lfloor\sqrt{TT_j}\rfloor$
13:    $j \leftarrow j+1$
14: **end while**
```

where $\delta' = \frac{\delta}{2|\mathcal{U}_T|\cdot(\log(\log N \log T))+4)}$ and $\tilde{\mu}_j(x) = z_{\mathcal{S}_j}^\top(x)\overline{v}_j$ is the *approximate* posterior mean computed based on the inducing set $\mathcal{S}_j$ (See Eqn. (5)). Since the posterior mean provides an estimate for the function values, the update condition is designed to eliminate all points at which the (estimated) function value is smaller than the current best estimate of the maximum value, upto an estimation error. Note that trimming is a deterministic procedure which ensures that all the agents and the server share a common value of $\mathcal{X}_{j+1}$. A detailed pseudocode of both the agent and the server side of the DUETS is provided in Algorithms 1 and 2 respectively.

The setup in DUETS where the server and each agent have access to shared randomness is similar to the class of *public coin protocols* that have been extensively studied in field of Information Theory. Such class of algorithms assume that agents and the server share a common source of randomness that is independent of the environment and comes at no communication cost between agents and the server Park et al. [2024], Acharya et al. [2020a,b]. Public coin protocols can concretely be implemented as an agent-server agreement on the generating seeds before the start of the algorithm, which takes at most $\mathcal{O}(1)$ communication cost. We stress that the source of shared randomness between the agent and the server is established before any interaction with the environment and thus cannot carry any information relevant to learning the reward function. It simply allows agents to follow an agreed on non-adaptive randomized strategy while querying the environment similar to the randomized strategy outlined in Acharya et al. [2020a] for choice of communication channels.

4 PERFORMANCE ANALYSIS

The following theorem characterizes the regret performance and communication cost of DUETS.

Theorem 4.1. *Consider the distributed kernel bandit problem described in Section 2. For a given $\delta \in (0,1)$, let the policy parameters of DUETS be such that $T_1 \geq \overline{M}/N$ and $p_0 = 72\log\frac{4N}{\delta}$. Then with probability at least $1-\delta$, the regret and communication cost incurred by DUETS satisfy the following relations:*

$$R_{\text{DUETS}} = \tilde{\mathcal{O}}(\sqrt{NT\gamma_{NT}}\log(T/\delta))$$
$$C_{\text{DUETS}} = \tilde{\mathcal{O}}(\gamma_{NT}).$$

Here, $\overline{M}$ is a constant that depends only on the kernel k and the domain $\mathcal{X}$ and is independent of N and T.[4]

[4]The constant $\overline{M}$ is the same as one in Lemma 4.3, which has been adopted from Salgia et al. [2023a]. We refer the reader to Salgia et al. [2023a] for an exact expression of the constant and additional related discussion.

As shown in above theorem, DUETS achieves order-optimal regret as it matches the lower bound established in Scarlett et al. [2017] upto logarithmic factors. DUETS is the *first algorithm* to close this gap to the lower bound in the distributed setup and achieve order-optimal regret performance. Moreover, DUETS incurs a communication cost that is sublinear in both T and N for all kernels. Furthermore, it can be much smaller that NT, depending upon the smoothness of the kernel. For example, using the bounds on information gain [Vakili et al., 2021b], we can show that the communication cost incurred by DUETS is $\mathcal{O}(\log^d(NT))$.

Proof. We provide a sketch of the proof of Theorem 4.1 here and refer the reader to Appendix B for a detailed proof.. The regret bound is obtained by first bounding the regret incurred by DUETS in each epoch j and then summing the regret across different epochs. In any epoch j, the agents take purely exploratory by uniformly sampling the region $\mathcal{X}_j$. Thus, to bound the regret incurred at any step during an epoch, we use the crude bound $\Delta_j := \sup_{x\in\mathcal{X}_j}(f(x^*) - f(x))$. Consequently, the regret during the j^{th} epoch, denoted by $R^{(j)}$, is upper bounded by $N\cdot\Delta_j T_j$. The epoch lengths are carefully chosen to balance the increase in epoch length with the decrease in Δ_j to obtain the tightest bound. These ideas are captured in the following lemmas.

Lemma 4.2. *Let $\Delta_j := \sup_{x\in\mathcal{X}_j} f(x^*) - f(x)$. Then, the following bound holds all epochs $j \geq 1$ with probability $1-\delta/2$.*

$$\Delta_j \leq 8\beta(\delta')\cdot\sup_{x\in\mathcal{X}_{j-1}}\sigma_j(x) + \frac{4B}{T},$$

where $\delta' = \frac{\delta}{2(\log(\log N+\log T)+4)|\mathcal{U}_T|}$ and $\mathcal{U}_T$ denotes the discretization defined in Assumption 2.3.

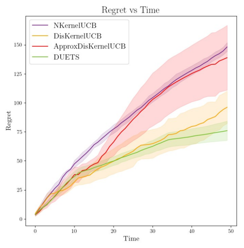
Lemma 4.3. *Salgia et al. [2023a] Let $\sigma_j^2(\cdot)$ denote the posterior variance corresponding to the set $\mathcal{D}_j$ obtained by sampling NT_j points uniformly at random from the domain $\mathcal{X}_j$. Then, for $T_1 \geq \overline{M}(\delta)/N$ and for any f satisfying Assumption 2.4, the following holds w.p. $1-\delta$ for all epochs $j\geq 1$:*

$$\sup_{x\in\mathcal{X}_j}\sigma_j^2(x) \leq C_{f,\mathcal{X}}\cdot\frac{\gamma_{NT_j}}{NT_j}.$$

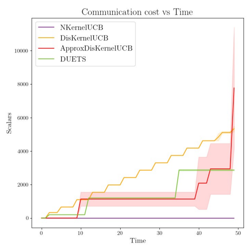
Here $C_{f,\mathcal{X}}$ denotes a constant that depends only on f and $\mathcal{X}$ and is independent of both N and T.

The bound on the communication cost follows directly from the following Lemmas 4.4 and 4.5 by noting that the communication cost in epoch j satisfies $\mathcal{O}(|\mathcal{S}_j|)$.

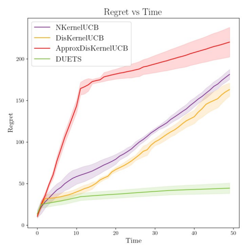
Lemma 4.4. *The total number of epochs in DUETS over a time horizon of T is at most* $\log(\log(\max\{N,T\}))+4$.



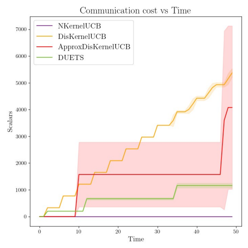
(a) $h_1(x)$



(b) $h_1(x)$



(c) $h_2(x)$



(d) $h_2(x)$

Figure 1: Cumulative regret Fig.(1a,1c) and communication cost Fig.(1b,1d) for all algorithms for the synthetic functions (h_1, h_2) averaged over 5 Monte Carlo runs. The shaded region represents error bars corresponding to one standard deviation. DUETS obtains a superior performance, both in terms of regret and communication cost, over other algorithms across all functions. Please refer to Appendix C for additional figures.

Lemma 4.5. *Let $\mathcal{S}_j$ denote the inducing set construct in j^{th} epoch, as outlined in Section 3. Then, for all epochs j the following holds with probability at least $1-\delta$,*

$$|\mathcal{S}_j| \leq C_{f,\mathcal{X}} \cdot (3 + \log(\log(\log N \log T)/\delta)) \cdot \gamma_{NT},$$

where $C_{f,\mathcal{X}}$ is same constant as the one in Lemma 4.3.

5 EMPIRICAL STUDIES

We perform several empirical studies to corroborate our theoretical findings. We compare the regret performance and communication cost of our proposed algorithm, DUETS, against three baseline algorithms — DisKernelUCB, ApproxDisKernelUCB and N-KernelUCB. The first two are distributed kernel bandits algorithms proposed in Li et al. [2022]. N-KernelUCB is a baseline algorithm considered in Li et al. [2022] where each agent locally runs the GP-UCB algorithm [Chowdhury and Gopalan, 2017] with no communication among the agents.

We compare the performance of all the four algorithm across four benchmark functions. The first two are synthetic functions $h_1, h_2 : \mathcal{B} \to \mathbb{R}$ considered in Li et al. [2022], where $\mathcal{B}$ denotes the unit ball centered at origin in $\mathbb{R}^{10}$. The functions are given by:

$$h_1(x) := \cos(3x^\top \theta^\star)$$
$$h_2(x) := (x^\top \theta^\star)^3 - 3(x^\top \theta^\star)^2 + 3(x^\top \theta^\star) + 3.$$

For both the functions $\theta^\star$ is randomly chosen from the surface of the unit ball $\mathcal{B}$. The other two functions are Branin [Azimi et al., 2012, Picheny et al., 2013] and Hartmann-4D [Picheny et al., 2013], which are commonly used benchmark functions for Bayesian Optimization. The Branin function is defined over $\mathcal{X} = [0,1]^2$ while the Hartmann-4D function is defined over $\mathcal{X} = [0,1]^4$.

We consider a distributed kernel bandit described in Section 2 with $N = 10$ agents. For all the experiments, we use the Squared Exponential kernel. The length scale was set to 0.2 for Branin and to 1 for all other functions. The observations were corrupted with zero mean Gaussian noise with a standard deviation of 0.2. The parameter D for ApproxDisKernelUCB and DisKernelUCB was set to 20 and 10 respectively. For DUETS , we set $T_1 = 2$ and $p_0 = 10$. The parameter β was selected using a grid search over $\{0.2, 0.5, 1, 2, 5\}$ for all the algorithms. All the algorithms were run for $T = 50$ time steps. We averaged the cumulative regret and the communication cost incurred by different algorithms over 5 Monte Carlo runs.

The cumulative regret (Fig. 1a, 1c) and the communication cost (Fig. 1b, 1d) incurred by different algorithms across the two synthetic functions (h_1, h_2) are shown Figure 1 (Please refer to Appendix C for additional figures). The shaded regions denotes error bars upto standard deviation on either side of the mean value. As evident from the plots, DUETS achieves a significantly lower regret as compared to all other algorithms consistently across benchmark functions. DUETS also incurs a smaller communication overhead as compared to other algorithms, corroborating our theoretical results.

6 CONCLUSION

We design the first algorithm for distributed kernel bandits that achieves order-optimal cumulative regret with a sub-linear communication cost. Crucial aspect of our approach is the uniform sampling based exploration strategy employed by agents. Data-independent exploration ensures that local exploration is synchronized with the centralized decision maker without frequent communication, thus significantly reducing the communication cost.

While our work establishes state-of-art regret and communication cost in non-contextual setting, ideas developed here are not directly applicable in the contextual setting. The design of efficient distributed algorithms in both stochastic and adversarial contextual settings remain an interesting avenue for future research.

7 Acknowledgments

The work of N. Pavlovic and Q. Zhao was supported in part by the National Science Foundation under Grant CCF-2419622 and the Government of Israel - Israeli Ministry of Defense, Mission to the USA.

References

Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems*, 2011. ISBN 9781618395993.

J. Acharya, C. L. Canonne, and H. Tyagi. Inference under information constraints ii: Communication constraints and shared randomness. *IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 66, NO. 12, DECEMBER 2020*, 2020a.

J. Acharya, C. L. Canonne, and H. Tyagi. Inference under information constraints i: Lower bounds from chi-square contraction. *IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 66, NO. 12, DECEMBER 2020*, 2020b.

S. Amani, T. Lattimore, A. György, and L. F. Yang. Distributed Contextual Linear Bandits with Minimax Optimal Communication Cost, 2022. URL `http://arxiv.org/abs/2205.13170`.

J. Azimi, A. Jalali, and X. Z. Fern. Hybrid batch bayesian optimization. In *Proceedings of the 29th International Conference on Machine Learning, ICML*, volume 2, pages 1215–1222, 2012. ISBN 9781450312851.

D. Calandriello, L. Carratino, A. Lazaric, M. Valko, and L. Rosasco. Gaussian Process Optimization with Adaptive Sketching: Scalable and No Regret. *Proceedings of Machine Learning Research*, 99:1–25, 2019.

R. Camilleri, J. Katz-Samuels, and K. Jamieson. High-Dimensional Experimental Design and Kernel Bandits. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

R. Chawla, A. Sankararaman, A. Ganesh, and S. Shakkottai. The Gossiping Insert-Eliminate Algorithm for Multi-Agent Bandits, 2020.

S. R. Chowdhury and A. Gopalan. On kernelized multi-armed bandits. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, volume 2, pages 1397–1422, 2017.

Z. Dai, B. K. H. Low, and P. Jaillet. Federated Bayesian optimization via Thompson sampling. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems*, volume 2020-Decem, 2020.

Y. Du, W. Chen, Y. Kuroki, and L. Huang. Collaborative Pure Exploration in Kernel Bandit. In *Proceedings of the 11th International Conference on Learning Representations, ICLR*, 2023.

A. Dubey and A. Pentland. Kernel methods for cooperative multi-agent contextual bandits. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, pages 2720–2730, 2020. ISBN 9781713821120.

A. Ghosh, A. Sankararaman, and K. Ramchandran. Adaptive Clustering and Personalization in Multi-Agent Stochastic Linear Bandits, 2021. URL `http://arxiv.org/abs/2106.08902`.

O. Hanna, L. Yang, and C. Fragouli. Learning from distributed users in contextual linear bandits without sharing the context. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, volume 35, pages 11049–11062, 2022.

R. Huang, W. Wu, J. Yang, and C. Shen. Federated Linear Contextual Bandits. In *Advances in Neural Information Processing Systems*, volume 32, pages 27057–27068, 2021. ISBN 9781713845393.

A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, pages 8571–8580, 2018.

S. G. Johnson. Saddle-point integration of "bump" functions. *arXiv preprint arXiv:1508.04376 30*, 2015.

N. Korda, B. Szorenyi, and S. Li. Distributed clustering of linear bandits in peer to peer networks. In *33rd International Conference on Machine Learning, ICML 2016*, volume 3, pages 1966–1980, 2016. ISBN 9781510829008.

P. Landgren, V. Srivastava, and N. E. Leonard. On distributed cooperative decision-making in multi-armed bandits. In *Proceedings of the European Control Conference, ECC*, pages 243–248, 2017. ISBN 9781509025916.

C. Li, H. Wang, M. Wang, and H. Wang. Communication Efficient Distributed Learning for Kernelized Contextual Bandits. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, 2022.

Z. Li and J. Scarlett. Gaussian process bandit optimization with few batches. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics, AISTATS*, 2022.

A. Mitra, H. Hassani, and G. Pappas. Exploiting Heterogeneity in Robust Federated Best-Arm Identification, 2021. URL `http://arxiv.org/abs/2109.05700`.

A. Mitra, H. Hassani, and G. J. Pappas. Linear Stochastic Bandits over a Bit-Constrained Channel, 2022.

H.-Y. Park, S.-H. Nam, and S.-H. Lee. Exactly optimal and communication- efficient private estimation via block designs. *IEEE Journal on selected areas in information theory , vol. 5, 2024*, 2024.

V. Picheny, T. Wagner, and D. Ginsbourger. A benchmark of kriging-based infill criteria for noisy optimization. *Structural and Multidisciplinary Optimization*, 48(3):607–626, 2013. ISSN 1615147X.

S. Salgia and Q. Zhao. Distributed linear bandits under communication constraints. In *Proceedings of the 40th International Conference on Machine Learning, ICML*, pages 29845–29875. PMLR, 2023.

S. Salgia, S. Vakili, and Q. Zhao. A domain-shrinking based Bayesian optimization algorithm with order-optimal regret performance. In *Proceedings of the 35th Annual Conference on Neural Information Processing Systems*, volume 34, 2021.

S. Salgia, S. Vakili, and Q. Zhao. Random exploration in bayesian optimization: Order-optimal regret and computational efficiency, 2023a.

S. Salgia, S. Vakili, and Q. Zhao. Collaborative learning in kernel-based bandits for distributed users. *IEEE Transactions on Signal Processing*, 71:3956–3967, 2023b.

A. Sankararaman, A. Ganesh, and S. Shakkottai. Social learning in multi agent multi armed bandits. *Proc. ACM Meas. Anal. Comput. Syst.*, 3(3), dec 2019.

J. Scarlett, I. Bogunovic, and V. Cehver. Lower Bounds on Regret for Noisy Gaussian Process Bandit Optimization. In *Conference on Learning Theory*, volume 65, pages 1–20, 2017.

S. Shahrampour, A. Rakhlin, and A. Jadbabaie. Multi-armed bandits in multi-agent networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2786–2790, 2017.

C. Shi and C. Shen. Federated Multi-Armed Bandits. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 9603–9611, 2021.

C. Shi, C. Shen, and J. Yang. Federated Multi-armed Bandits with Personalization, 2021. URL `http://arxiv.org/abs/2102.13101`.

N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning, ICML*, pages 1015–1022, 2010.

S. Vakili, N. Bouziani, S. Jalali, A. Bernacchia, and D.-s. Shiu. Optimal order simple regret for Gaussian process bandits. In *Proceedings of the 35th Annual Conference on Neural Information Processing Systems*, 2021a.

S. Vakili, K. Khezeli, and V. Picheny. On information gain and regret bounds in Gaussian process bandits. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics, AISTATS*, 2021b.

S. Vakili, J. Scarlett, D.-s. Shiu, and A. Bernacchia. Improved convergence rates for sparse approximation methods in kernel-based learning. In *Proceedings of the 39th International Conference on Machine Learning, ICML*, pages 21960–21983. PMLR, 2022.

M. Valko, N. Korda, R. Munos, I. Flaounas, and N. Cristianini. Finite-time analysis of kernelised contextual bandits. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence, UAI*, pages 654–663, 2013.

Y. Wang, J. Hu, X. Chen, and L. Wang. Distributed Bandit Learning: Near-Optimal Regret with Efficient Communication. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.

V. Wild, M. Kanagawa, and D. Sejdinovic. Connections and equivalences between the nyström method and sparse variational gaussian processes, 2021.

Z. Zhu, J. Zhu, J. Liu, and Y. Liu. Federated Bandit: A Gossiping Approach. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 5 (1):1–29, 2021.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Yes.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries.
   Yes.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results.
   Yes
   (b) Complete proofs of all theoretical results.
   Yes.
   (c) Clear explanations of any assumptions.
   Yes

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL).
   Yes.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen).
   Not Applicable
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times).
   Yes.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider).
   Yes.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets.
   Not Applicable.
   (b) The license information of the assets, if applicable. Not Applicable.
   (c) New assets either in the supplemental material or as a URL, if applicable.
   Not Applicable.
   (d) Information about consent from data providers/curators.
   Not Applicable.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content.
   Not Applicable.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots.
   Not Applicable
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable.
   Not Applicable
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation.
   Not Applicable

A Additional details about DUETS

Below we outline the pseudo-code of DUETS from the server side.

B Proof of Theorem 4.1

In this section, we provide a detailed proof of Theorem 4.1. For the regret bound, we first bound the regret incurred by DUETS in each epoch j and then sum it across different epochs to obtain a bound on the overall cumulative regret. We first prove the theorem assuming the results from Lemmas 4.2, 4.3 and 4.4 and then separately prove the lemmas.

Consider any epoch $j \geq 1$ and let $R^{(j)}$ denote the regret incurred by DUETS in this epoch. Since the agents take purely exploratory actions by uniform sampling points from the current set, we have the following crude bound $R^{(j)} \leq \Delta_j \cdot NT_j \cdot M_f$, where $\Delta_j := \sup_{x \in \mathcal{X}_j}(f(x^*) - f(x))$. The term $NT_j \cdot M_f$ corresponds to number of points sampled during the epoch as we sample each connected component of $\mathcal{X}_j$, of which there are at most M_f, NT_j times. For $j = 1$, we use the trivial bound,

$$\Delta_1 = \sup_{x \in \mathcal{X}}(f(x^*) - f(x)) \leq 2 \sup_{x \in \mathcal{X}} f(x) \leq 2B,$$

which gives us $R^{(1)} \leq 2B \cdot NT_1 \cdot M_f$. On invoking Lemma 4.2 for $j > 1$ we obtain,

$$\begin{aligned} R^{(j)} &\leq \Delta_j \cdot NT_j \cdot M_f \\ &\leq NT_j \cdot M_f \cdot \left(8\beta(\delta') \cdot \left(\sup_{x \in \mathcal{X}_{j-1}} \sigma_{j-1}(x) \right) + \frac{4B}{T} \right), \end{aligned}$$

where $\delta' = \dfrac{\delta}{2(\log\log NT + 4)|\mathcal{U}_T|}$. Using Lemma 4.3, we can further bound this expression as

$$\begin{aligned} R^{(j)} &\leq \Delta_j \cdot NT_j \cdot M_f \\ &\leq NT_j \cdot M_f \cdot \left(8\beta(\delta') \cdot C_{f,\mathcal{X}} \cdot \sqrt{\frac{\gamma_{NT_{j-1}}}{NT_{j-1}}} + \frac{4B}{T} \right) \\ &\leq M_f \cdot \left(8C_{f,\mathcal{X}}^{1/2} \cdot \beta(\delta') \cdot \sqrt{NT\gamma_{NT_{j-1}}} + \frac{4BNT_j}{T} \right) \\ &\leq M_f \cdot \left(8C_{f,\mathcal{X}}^{1/2} \cdot \beta(\delta') \cdot \sqrt{NT\gamma_{NT}} + \frac{4BNT_j}{T} \right). \end{aligned}$$

In the third line, we used the inequality $\frac{T_j}{\sqrt{T_{j-1}}} \leq \sqrt{T}$, which follows from the definition of T_j. In the last line, we used the fact that γ_m is an increasing function of m. Thus, if J denotes an upper bound on the number of epochs, we can write:

$$\begin{aligned} \sum_{j=1}^{J} R^{(j)} &\leq 2BM_f \cdot NT_1 + \sum_{j=2}^{J} M_f \cdot \left(8C_{f,\mathcal{X}}^{1/2} \cdot \beta(\delta') \cdot \sqrt{NT\gamma_{NT}} + \frac{4BNT_j}{T} \right) \\ &\leq 2BM_f \cdot NT_1 + J \cdot \left(8C'_{f,\mathcal{X}} \cdot \beta(\delta') \cdot \sqrt{NT\gamma_{NT}} \right) + \frac{4BNM_f}{T} \sum_{j=1}^{J} T_j \\ &\leq 2BM_f \cdot NT_1 + J \cdot \left(8C'_{f,\mathcal{X}} \cdot \beta(\delta') \cdot \sqrt{NT\gamma_{NT}} \right) + 4BNM_f. \end{aligned} \tag{10}$$

We next optimize the length of the first epoch T_1 in order to achieve order optimal regret. DUETS achieves order optimal regret for $N \leq \max(T, \gamma_{NT})$.

If $N < T$ we can choose $T_1 = \sqrt{\frac{T}{N}} + \overline{M}(\delta^{'})$ where $\delta^{'} = \frac{\delta}{2(\log\log NT+4)}$. Left hand side of equation (10) can now be written as $\widetilde{\mathcal{O}}(\sqrt{NT\gamma_{NT}}\beta(\delta^{'})) \equiv \widetilde{\mathcal{O}}\left(\sqrt{NT\gamma_{NT}}\left(\log\frac{T}{\delta}\right)\right)$.

If $N \leq \gamma_{NT}$ we can fix $T_1 = \sqrt{T} + \overline{M}(\delta')$. We have $NT_1 \leq \widetilde{O}(\sqrt{NT\gamma_{NT}})$ and the left hand-side is once again $\widetilde{\mathcal{O}}\left(\sqrt{NT\gamma_{NT}}\left(\log\frac{T}{\delta}\right)\right)$.

Note that by Lemma 4.4, J is upper bounded by $\log(\log N \log T) + 4$ and is thus $\widetilde{O}(1)$.

Before moving onto the proofs of Lemmas 4.2 and 4.4, we state two auxiliary lemmas that will be useful for our analysis.

Definition B.1. Let $\mathcal{D} = \{x_1, x_2, \ldots, x_m\} \subset \mathcal{X}$ be a collection m points and $\mathcal{S}$ be any subset of $\mathcal{D}$. Let $\sigma^2_{\mathcal{D}}(\cdot)$ denote the posterior variance corresponding to the points in $\mathcal{D}$ and $\tilde{\sigma}^2_{\mathcal{S}}(\cdot)$ denote the *approximate* posterior computed based on the points in $\mathcal{S}$. We call $\mathcal{S}$ to be an ε-accurate inducing set if the following relations are true for all $x \in \mathcal{X}$.

$$\frac{1-\varepsilon}{1+\varepsilon} \cdot \tilde{\sigma}^2_{\mathcal{S}}(x) \leq \sigma^2_{\mathcal{D}}(x) \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot \tilde{\sigma}^2_{\mathcal{S}}(x).$$

Lemma B.2 (Adapted from Calandriello et al. [2019])**.** *Let $\mathcal{D} = \{x_1, x_2, \ldots, x_m\} \subset \mathcal{X}$ be a collection m points and $\mathcal{S}$ be a random subset of $\mathcal{D}$ constructed by including each point with probability p, independent of other points. Then $\mathcal{S}$ is an ε-accurate inducing set with probability $1 - 4m\exp\left(-\frac{3p\varepsilon^2}{8\sigma^2_{\max}}\right)$, where $\sigma^2_{\max} = \sup_{x\in\mathcal{X}} \sigma^2_{\mathcal{D}}(x)$.*

Lemma B.3. *Let DUETS be run with a choice of $p_0 = 72\log(4NT/\delta')$. Then, for all epochs $j \geq 1$, the global inducing set $\mathcal{S}_j$ is 0.5-accurate with probability $1-\delta$.*

Proof. The statement is an immediate consequence of Lemma B.2 with the given choice of parameter p_0. □

We are now ready to prove Lemmas 4.2 and 4.4.

B.1 Proof of Lemma 4.2

Consider any epoch $j \geq 2$ and let $x \in \mathcal{X}_j$. Let $\Delta(x) := f(x^*) - f(x)$. We will obtain a bound on $\Delta(x)$ for any general x in order establish the bound on Δ_j. Using the discretization from Assumption 2.3 for $\mathcal{X}_j$, we obtain,

$$\begin{aligned}\Delta(x) &= f(x^*) - f(x)\\ &\leq f(x^*) - f([x^*]_{\mathcal{U}_T}) + f([x^*]_{\mathcal{U}_T}) - (f(x) - f([x]_{\mathcal{U}_T})) - f([x]_{\mathcal{U}_T})\\ &\leq f([x^*]_{\mathcal{U}_T}) - f([x]_{\mathcal{U}_T}) + \frac{2B}{T}.\end{aligned}$$

Using the result from Salgia et al. [2023b], we obtain the following high probability bound that holds with probability $1-\delta$:

$$\begin{aligned}\Delta(x) &\leq f([x^*]_{\mathcal{U}_T}) - f([x]_{\mathcal{U}_T}) + \frac{2B}{T}\\ &\leq \tilde{\mu}_j([x^*]_{\mathcal{U}_T}) + \beta(\delta')\tilde{\sigma}_j([x^*]_{\mathcal{U}_T}) - \tilde{\mu}_j([x]_{\mathcal{U}_T}) + \beta(\delta')\tilde{\sigma}_j([x]_{\mathcal{U}_T}) + \frac{2B}{T}\\ &\leq \tilde{\mu}_j(x^*) - \tilde{\mu}_j(x) + \beta(\delta')\tilde{\sigma}_j([x^*]_{\mathcal{U}_T}) + \beta(\delta')\tilde{\sigma}_j([x]_{\mathcal{U}_T}) + \frac{4B}{T},\end{aligned}$$

where we again used Assumption 2.3 in the last step. We claim that $x^* \in \mathcal{X}_{j-1}$ for all $j \geq 2$. Assuming this claim this true, we can bound the above expression as

$$\begin{aligned}\Delta(x) &\leq \sup_{x\in\mathcal{X}_{j-1}} \tilde{\mu}_j(x') - \tilde{\mu}_j(x) + \beta(\delta')\tilde{\sigma}_j([x^*]_{\mathcal{U}_T}) + \beta(\delta')\tilde{\sigma}_j([x]_{\mathcal{U}_T}) + \frac{4B}{T}\\ &\leq 2\beta(\delta')\sigma_{j,\max} + \beta(\delta')\tilde{\sigma}_j([x^*]_{\mathcal{U}_T}) + \beta(\delta')\tilde{\sigma}_j([x]_{\mathcal{U}_T}) + \frac{4B}{T},\end{aligned}$$

where we used the update condition (Eqn. (9)) in the second step. Since $\mathcal{S}_j$ is 0.5-accurate (Lemma B.3), we have $\tilde{\sigma}^2_j(x) \leq 3\sigma^2_j(x) \leq 3\sigma^2_{j,\max}$. On plugging this back into the above equation, we obtain,

$$\Delta(x) \leq 8\beta(\delta')\sigma_{j,\max} + \frac{4B}{T}.$$

The statement of the lemma follows by $\Delta_j = \sup_{x\in\mathcal{X}_j} \Delta(x)$.

We prove our claim $x^* \in \mathcal{X}_j$ for all $j \geq 1$ using induction. Clearly, $x^* \in \mathcal{X}_1 = \mathcal{X}$, by definition. Assume $x^* \in \mathcal{X}_{j-1}$ for some $j \geq 2$. Fix an arbitrary $x \in \mathcal{X}_{j-1}$, from the confidence bound lemma we have:

$$\mu_{j-1}(x) - \mu_{j-1}(x^*) \leq (f(x) - f(x^*)) + \beta(\delta^{'})(\sigma_j(x) + \sigma_j(x^*)) \leq 2\sigma_{j-1.\max}(x),$$

where the second inequality follows as $f(x) \leq f(x^*)$. As the inequality holds $\forall x \in \mathcal{X}_{j-1}$ we must have:

$$\sup_{x\in\mathcal{X}_{j-1}} \mu_{j-1}(x) - \mu_{j-1}(x^*) \leq 2\sigma_{j-1.\max}(x)$$

and thus indeed $x \in \mathcal{X}_j$.

B.2 Proof of Lemma 4.4

We define $E(s) := \min\{j : T_j \geq T/4 \mid T_1 = s\}$. Note that T_j is an increasing function of j. Since $T_{E(s)} \geq T/4$, we can conclude that $E(s) + 4$ is an upper bound on the number of epochs. Thus, we focus on bounding $E(s)$. We first show that $E(s)$ is a non-decreasing function of s.

To that effect, we claim that for $j \geq 2$ the epoch lengths satisfy the relation $T_j \geq T^{1-2^{-j+1}} \cdot T_1^{2^{-j+1}}$. This relation follows immediately using induction. For the base case, note that $T_2 \geq T^{1/2} \cdot T_1^{1/2}$, by definition. Assume that the relation holds for $j-1$. Thus,

$$T_j \geq T^{1/2} \cdot T_{j-1}^{1/2} \geq T^{1/2} \cdot T^{1-2^{-(j-1)+1-1}} \cdot T_1^{2^{-(j-1)+1-1}} \geq T^{1-2^{-j+1}} \cdot T_1^{2^{-j+1}}. \tag{11}$$

Since T_j's are lower bounded by an increasing function of T_1, the number of epochs $E(s)$ is a non-increasing function of s. Since $T_1 \geq \frac{T}{N}$, $E\left(\frac{T}{N}\right)$ is an upper bound on the number of epochs for all choices of T_1.

Let $j^* = \max\{\log(\log(T)), \log(\log(N))\}$. Using the above relation for T_j from Eqn. (11) and the lower bound on T_1, we have,

$$T_{j^*} \geq T \cdot N^{-2^{1-j}} = T \cdot \left(2^{-\frac{\log N}{2^j}}\right)^2 \geq T \cdot 2^{-2}$$

We can hence conclude that $T_{j^*} \geq T/4$, which implies that $E(T_1) \leq j^*$ for all permissible choices of T_1. Consequently, the number of epochs are bounded as $\log(\log(\max\{N, T\})) + 4$.

B.3 Proof of Lemma 4.5

For all epochs $j \geq 1$, recall that the inducing set is constructed by including each point from $\mathcal{D}_j$ with probability p_j, independent of other points. Thus, $|\mathcal{S}_j|$ is a binomial random variable with parameters $|\mathcal{D}_j| = NT_j$ and p_j. Using the Chernoff bound for Binomial random variables, we can conclude that

$$\Pr(|\mathcal{S}_j| > (1+\varepsilon)NT_jp_j) \leq \exp\left(-\frac{\varepsilon^2 NT_jp_j}{2+\varepsilon}\right).$$

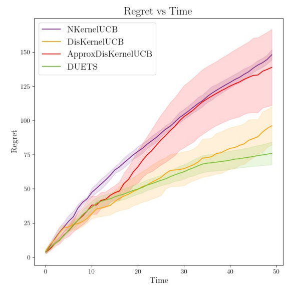
Invoking the bound with $\varepsilon = 2 + \log(1/\delta')$, with $\delta' = \delta/(\log\log(NT) + 4)$ yields that the following relation holds with probability $1 - \delta'$:

$$\begin{aligned}
|\mathcal{S}_j| &\leq (3 + \log(1/\delta')) \cdot NT_jp_j \\
&\leq (3 + \log(1/\delta')) \cdot NT_j \cdot p_0\sigma_{j,\max}^2 \\
&\leq (3 + \log(1/\delta')) \cdot NT_jp_0 \cdot C_{f,\mathcal{X}} \cdot \frac{\gamma_{NT_j}}{NT_j} \\
&\leq (3 + \log(1/\delta'))p_0\gamma_{NT},
\end{aligned}$$

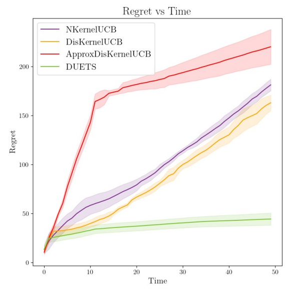
where we used Lemma 4.3 in the third step and monotonicity of γ_m in the last step. On taking a union bound over all epochs and using the bound on the number of epochs from Lemma 4.4, we conclude that for all epochs j, $|\mathcal{S}_j| = \tilde{\mathcal{O}}(\gamma_{NT})$ with probability $1 - \delta$.

C Additional details about experiments

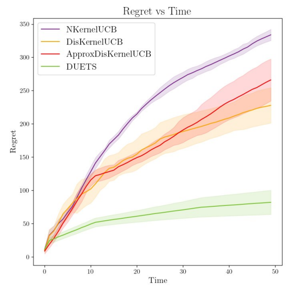
In this section, we provide additional details about our empirical studies along with results on benchmark functions.



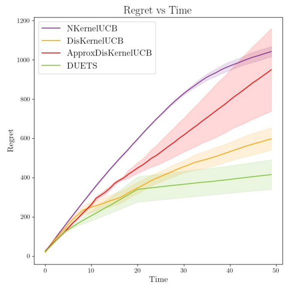
(a) $h_1(x)$



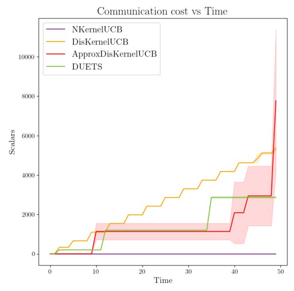
(b) $h_2(x)$



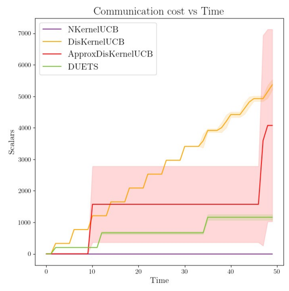
(c) Branin



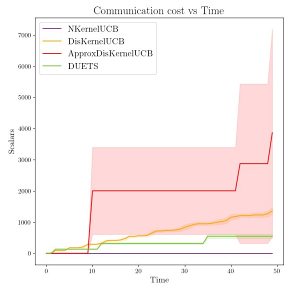
(d) Hartmann-4D



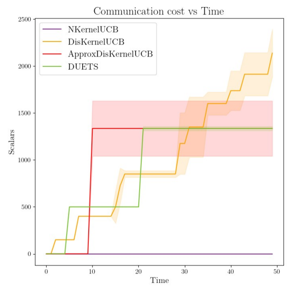
(e) $h_1(x)$



(f) $h_2(x)$



(g) Branin



(h) Hartmann-4D

Figure 2: Cumulative regret (Fig. (2a-2d) and communication cost (2e-2h) for all algorithms across different benchmark functions averaged over 5 Monte Carlo runs. The shaded region represents error bars corresponding to one standard deviation. DUETS obtains a superior performance, both in terms of regret and communication cost, over other algorithm across all functions.

We consider a distributed kernel bandit described in Section 2 with $N = 10$ agents. For all the experiments, we use the Squared Exponential kernel. The length scale was set to 0.2 for Branin and to 1 for all other functions. The observations were corrupted with zero mean Gaussian noise with a standard deviation of 0.2. The parameter D for ApproxDisKernelUCB and DisKernelUCB was set to 20 and 10 respectively. For DUETS , we set $T_1 = 2$ and $p_0 = 10$. The parameter β was selected using a grid search over $\{0.2, 0.5, 1, 2, 5\}$ for all the algorithms. All the algorithms were run for $T = 50$ time steps. We averaged the cumulative regret and the communication cost incurred by different algorithms over 5 Monte Carlo runs.

The cumulative regret incurred by different algorithms across the different benchmark function are shown in the top row of Figure 2. The bottom row consists of the corresponding plots for the communication cost incurred by the different algorithm. The shaded regions denotes error bars upto standard deviation on either side of the mean value. As evident from the plots, DUETS achieves a significantly lower regret as compared to all other algorithms consistently across benchmark functions. DUETS also incurs a smaller communication overhead as compared to other algorithms, corroborating our theoretical results.

All simulations were run on an Intel(R) Xeon(R) E-2176M CPU@ 2.70GHz with 6 cores with no GPU's. Runtime of 5 Monte-Carlo-simulations for all 4 algorithms and over all benchmarks took roughly 6 hours of CPU time. Run-time of DUETS was 2.8s, averaged over 10 agents and 4 benchmarks. Notably the run-time of DUETS is faster by a factor of 60 compared to other algorithms we used for comparison.