Chapter 14

Federated Sequential Decision Making: Bayesian Optimization, Reinforcement Learning, and Beyond

Zhongxiang Dai*[a], Flint Xiaofeng Fan*[a], Cheston Tan[c], Trong Nghia Hoang[d], Bryan Kian Hsiang Low[a], and Patrick Jaillet[b]

[a]*National University of Singapore, Singapore,* [b]*Massachusetts Institute of Technology, MA, USA,* [c]*Institute for Infocomm Research, A*STAR, Singapore,* [d]*Washington State University, WA, USA*

ABSTRACT

Federated learning (FL) in its classic form involves the collaborative training of supervised learning models (e.g., neural networks) among multiple agents/clients. However, in addition to supervised learning, many other machine learning tasks which are inherently *sequential decision-making* problems, such as *Bayesian optimization* (BO) and *reinforcement learning* (RL), also find important applications in the federated setting. For example, the crucial problem of hyperparameter tuning of neural networks in the federated setting calls for algorithms for *federated BO*; collaborative clinical treatment recommendation among multiple hospitals is a natural application for *federated RL*. However, the extension of these classic sequential decision-making algorithms into the federated setting is faced with immense challenges. Firstly, these algorithms (e.g., BO and RL) have to be adapted to satisfy the core principles of FL. For example, consistent with the requirement of FL, the raw data (e.g., the history of observations in BO and the trajectories in RL) of every agent can never be shared with the other agents. Next, it is challenging to preserve the rigorous theoretical guarantees of these classic sequential decision-making algorithms (e.g., the sub-linear regret upper bound of classic BO algorithms and the sample complexity of classic policy gradient algorithms for RL) and at the same time consistently improve their empirical performances by leveraging the federation of multiple agents. In this regard, a number of recent works have tackled these challenges and hence introduced federated versions of classic sequential decision-making algorithms (e.g., federated BO and federated RL algorithms) which satisfy the core principles of FL and are both theoretically grounded and practically effective. In light of these recent advances, this chapter discusses federated sequential decision-making problems with a focus on recent representative works on federated BO and federated RL, and describes open problems and potential future directions in these areas.

* denotes equal contribution.

KEYWORDS
Federated Sequential Decision-Making, Federated Bayesian Optimization, Federated Reinforcement Learning, Federated Bandits, Federated Hyperparameter Tuning

1.1 INTRODUCTION

Classic federated learning (FL) is designed for supervised learning, i.e., multiple agents/clients collaborate to train a supervised learning model such as a neural network (NN) [54] or a decision tree-based model [44, 45]. However, in addition to supervised learning, many other machine learning methods which are *sequential decision-making* problems in nature, such as the celebrated methods of *Bayesian optimization* (BO) [28] and *reinforcement learning* (RL) [71], also find important applications in the federated setting. For example, BO has been the most popular method for tuning the hyperparameters of ML models, and hence hyperparameter tuning of ML models in the federated setting naturally calls for algorithms for *federated BO* (FBO); RL has been widely adopted for clinical decision support, and therefore collaborative clinical treatment recommendation among multiple hospitals is a natural and important application for *federated RL* (FRL). Therefore, extending these classic sequential decision-making algorithms (such as BO and RL) into the federated setting holds considerable promise for more widespread applications of FL.

However, the extension of these classic algorithms into the federated setting is non-trivial and faced with significant challenges. First of all, these classic sequential decision-making algorithms need to be modified to satisfy the core principles of FL. As a prime example, an important principle in FL is that the raw data of an agent can never be transmitted [35]. Similarly, during federated sequential decision-making, the raw data of an agent, such as the history of observations in BO and the raw trajectories in RL, must also be retained by the agent and hence never shared with others. This requirement, as well as other requirements stemming from the core principles of FL, may necessitate non-trivial problem-dependent algorithmic designs. Moreover, when modifying these sequential decision-making algorithms for the federated setting, it is challenging to preserve their rigorous theoretical guarantees (e.g., the sub-linear regret upper bound of classic BO algorithms and the sample complexity of classic policy gradient algorithms for RL) and at the same time consistently improve their empirical performances by exploiting the federation of multiple agents. In recent years, a number of works have tackled these challenges and hence introduced federated versions of classic sequential decision-making algorithms, such as FBO [14, 15] and FRL [26] algorithms.

The works of [15] and [14] have proposed FBO algorithms. The work of [15] has introduced the first FBO algorithm, named *federated Thompson sampling* (FTS), which addressed several important challenges in FBO. Specifically, the

FTS algorithm is free from the requirement to transmit the observations of BO, requires a small number of parameters to be exchanged, and is theoretically guaranteed to be no-regret despite agent heterogeneity [15]. The more recent work of [14] has extended the FTS algorithm of [15] to incorporate a rigorous privacy guarantee and to further improve its practical performance via the method of distributed exploration. Regarding FRL, the recent work of [26] has introduced the first FRL framework with a theoretically guaranteed convergence. Specifically, building on the classic policy gradient algorithm, [26] only requires the agents to exchange their policy gradients instead of their raw trajectories and has achieved low sample complexity by leveraging variance-reduced optimization techniques. Moreover, [26] has also achieved theoretically guaranteed robustness against Byzantine agents (i.e., caused by random failures or adversarial attacks), which is another important consideration in FL. These recent works on FBO [14, 15] and FRL [26] have been shown to be both theoretically grounded and practically effective, and we will discuss them in more detail in Sections 1.2 and 1.3, respectively. In addition to FBO and FRL, another line of recent works has extended *multi-armed bandits* [40], which is another classic sequential decision-making problem, to the federated setting and hence introduced federated bandits [16, 22, 66]. These recent works on federated bandits mostly focus on the theoretical perspective, and we will discuss them in Section 1.4.3.

In the remainder of this chapter, we will separately discuss FBO (Section 1.2) and FRL (Section 1.3), including their background, representative existing works, algorithms/frameworks, as well as theoretical and empirical results. Subsequently, in Section 1.4, we will briefly review other recent related works on FBO (Section 1.4.1), FRL (Section 1.4.2) and federated bandits (Section 1.4.3). Lastly, we will discuss open problems and potential future directions in the area of federated sequential decision-making (Section 1.5).

1.2 FEDERATED BAYESIAN OPTIMIZATION

In this section, we firstly present some technical background on BO (Section 1.2.1) and FBO (Section 1.2.2). Next, we give an overview of the representative existing works on FBO [14, 15] (Section 1.2.3), and then discuss their algorithms (Section 1.2.4) as well as their theoretical and empirical results (Section 1.2.5).

1.2.1 Background on Bayesian Optimization

Bayesian optimization (BO) aims to use sequential queries to maximize an objective function $f : \mathcal{X} \to \mathbb{R}$, i.e., to find $\mathbf{x}^* \in \arg\max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$, in which $\mathcal{X} \subset \mathbb{R}^d$ is a discrete subset of the $d-$dimensional domain.[1] The function f is usu-

[1] We assume $\mathcal{X}$ to be discrete for simplicity, but the extension to a continuous domain is straightforward using well known techniques [14].

ally black-box (i.e., non-differentiable and hence only available through queries) and costly to evaluate. A typical example is hyperparameter tuning for deep neural networks (DNNs), in which $\mathbf{x} \in \mathcal{X}$ is a hyperparameter configuration for training a DNN (e.g., the learning rate, regularization parameter, etc.) and $f(\mathbf{x})$ represents the validation accuracy obtained after training the DNN using the hyperparameter configuration $\mathbf{x}$. Specifically, in iteration $t = 1, \ldots, T$ of BO, an input $\mathbf{x}_t \in \mathcal{X}$ is selected and queried, yielding an output observation: $y_t \triangleq f(\mathbf{x}_t) + \zeta$ where ζ is sampled from a zero-mean Gaussian noise with a variance of σ^2: $\zeta \sim \mathcal{N}(0, \sigma^2)$. To select the input queries $\mathbf{x}_t$'s intelligently, BO uses a *Gaussian process* (GP) [60] as a surrogate to model the function f.

A GP $\mathcal{GP}(\mu(\cdot), k(\cdot,\cdot))$ is characterized by its mean function μ and kernel function k. We assume w.l.o.g. that $\mu(\mathbf{x}) = 0$ and $k(\mathbf{x}, \mathbf{x}') \leq 1, \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}$, and we mainly focus on the widely used *squared exponential* (SE) kernel here [14, 15]. In iteration $t + 1$ of BO, given the first t observed input-output pairs, the GP posterior is given by $\mathcal{GP}(\mu_t(\cdot), \sigma_t^2(\cdot,\cdot))$ where

$$\begin{aligned} \mu_t(\mathbf{x}) &:= \mathbf{k}_t(\mathbf{x})^\top (\mathbf{K}_t + \lambda \mathbf{I})^{-1} \mathbf{y}_t \\ \sigma_t^2(\mathbf{x}, \mathbf{x}') &:= k(\mathbf{x}, \mathbf{x}') - \mathbf{k}_t(\mathbf{x})^\top (\mathbf{K}_t + \lambda \mathbf{I})^{-1} \mathbf{k}_t(\mathbf{x}') \end{aligned} \tag{1.1}$$

where $\mathbf{k}_t(\mathbf{x}) := (k(\mathbf{x}, \mathbf{x}_\tau))_{\tau=1,\ldots,t}^\top$ is a $t-$dimensional column vector, $\mathbf{y}_t := (y_\tau)_{\tau=1,\ldots,t}^\top$ is also a $t-$dimensional column vector, $\mathbf{K}_t := (k(\mathbf{x}_\tau, \mathbf{x}_{\tau'}))_{\tau,\tau'=1,\ldots,t}$ is a $t \times t-$dimensional matrix and $\lambda > 0$ is a regularization parameter [10]. In iteration $t + 1$, the GP posterior (1.1) is used to calculate an *acquisition function*, which is then maximized to select the next query $\mathbf{x}_{t+1}$. For example, the classic *Thompson sampling* (TS) [10] algorithm firstly samples a function f_{t+1} using the GP posterior (1.1) and then uses it as the acquisition function to chooses $\mathbf{x}_{t+1} = \arg\max_{\mathbf{x} \in \mathcal{X}} f_{t+1}(\mathbf{x})$. BO algorithms are usually analyzed in terms of *regrets* [70]. A hallmark for well-performing BO algorithms is to be asymptotically *no-regret*, which requires the *cumulative regret* $R_T := \sum_{t=1}^T (f(\mathbf{x}^*) - f(\mathbf{x}_t))$ to grow sub-linearly so that the simple regret $S_T := \min_{t=1\ldots,T} (f(\mathbf{x}^*) - f(\mathbf{x}_t)) \leq R_T/T$ goes to 0 asymptotically.

To reduce the computational cost of GP posterior inference (1.1), *random Fourier features* (RFFs) [59] is commonly adopted to approximate the kernel function k using M-dimensional random features $\boldsymbol{\phi}$: $k(\mathbf{x}, \mathbf{x}') \approx \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$. RFFs offers a high-probability guarantee on the approximation quality, i.e., we have that $\sup_{\mathbf{x}, \mathbf{x}' \in \mathcal{X}} |k(\mathbf{x}, \mathbf{x}') - \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')| \leq \varepsilon$, in which $\varepsilon = O(M^{-1/2})$ [59]. Of note, RFFs makes it particularly convenient to approximately sample a function from the GP posterior (1.1). Specifically, define $\boldsymbol{\Phi}_t := (\boldsymbol{\phi}(\mathbf{x}_\tau))_{\tau \in [t]}^\top$ (i.e., a $t \times M$-dimensional matrix), $\boldsymbol{\Sigma}_t := \boldsymbol{\Phi}_t^\top \boldsymbol{\Phi}_t + \lambda \mathbf{I}$, and $\boldsymbol{\nu}_t := \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\Phi}_t^\top \mathbf{y}_t$. To sample a function $\widetilde{f}$ from the GP posterior with the approximate kernel $\widetilde{k}(\mathbf{x}, \mathbf{x}') \approx \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$, we only need to sample an $M-$dimensional vector

$$\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{\nu}_t, \lambda \boldsymbol{\Sigma}_t^{-1}) \tag{1.2}$$

and then set $\widetilde{f}(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\top \omega, \forall \mathbf{x} \in \mathcal{X}$. Refer to [15] for more details on RFF approximations of GP.

1.2.2 Background on Federated Bayesian Optimization

Here we present the problem setting of *federated BO* (FBO), which follows the works of [14] and [15]. FBO involves a central server and N agents/clients $\mathcal{A}_1, \ldots, \mathcal{A}_N$. Every agent $\mathcal{A}_n$ attempts to maximize its own objective function $f^n : \mathcal{X} \to \mathbb{R}$, i.e., to find $\mathbf{x}^n_* \in \arg\max_{\mathbf{x}\in\mathcal{X}} f^n(\mathbf{x})$, by querying $\mathbf{x}^n_t$ and observing $y^n_t, \forall t = 1, \ldots, T$. As a representative motivating example, N mobile phone users (agents) who use DNNs for next-word prediction in a smart keyboard application may wish to collaboratively tune the hyperparameters of their DNNs. In this application, agent (mobile phone user) $\mathcal{A}_n$ sequentially queries the hyperparameter configurations $\mathbf{x}^n_t \in \mathcal{X}, \forall t = 1 \ldots, T$ in order to maximize its validation accuracy denoted by the function $f^n : \mathcal{X} \to \mathbb{R}$. As another example, a hospital can use BO to select the patients to perform a medical test to assess the possibility of readmission [81], and multiple hospitals may wish to collaborate to achieve better patient selection strategies. In this case, the input query $\mathbf{x}^n_t \in \mathcal{X}$ selected by hospital $\mathcal{A}_n$ in iteration t corresponds to the features representing the selected patient, and the corresponding value of the objective function $f^n(\mathbf{x}^n_t)$ represents the test score for the selected patient $\mathbf{x}^n_t$.

As a common ground for the collaboration among different agents, we assume that all participating agents share the same set of random features $\boldsymbol{\phi}(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$. Similar to BO discussed above, in FBO, every agent $\mathcal{A}_n$ aims to minimize its own cumulative regret: $R^n_T \triangleq \sum_{t=1}^T (f^n(\mathbf{x}^n_*) - f^n(\mathbf{x}^n_t))$. Without loss of generality, when presenting the theoretical results in Section 1.2.5, we focus on the perspective of agent $\mathcal{A}_1$. That is, we derive an upper bound on the cumulative regret of agent $\mathcal{A}_1$ denoted as R^1_T. We characterize the similarity between agents $\mathcal{A}_1$ and $\mathcal{A}_n$ by $d_n := \max_{\mathbf{x}\in\mathcal{X}} |f^1(\mathbf{x}) - f^n(\mathbf{x})|$, such that $d_1 = 0$ and a smaller d_n indicates a larger degree of similarity (or equivalently, a smaller degree of heterogeneity) between $\mathcal{A}_1$ and $\mathcal{A}_n$. In the theoretical analysis, we assume that the objective functions of all agents have a bounded norm induced by the *reproducing kernel Hilbert space* (RKHS) associated with the kernel k, i.e., $\left\|f^n\right\|_k \leq B, \forall n \in [N]$. This essentially assumes that the objective functions are smooth and is a common assumption in the analysis of BO algorithms [10].

Core Principles and Challenges of FBO. Since FBO needs to follow the core principles of FL, it also inherits a number of important challenges of the federated setting. Firstly, the raw data (e.g., the history of queried hyperparameter configurations and observed validation accuracies during BO) of every agent can never be shared with others. Secondly, the heterogeneity among different agents is another crucial challenge of FBO. For example, since different mobile phone users may have distinct typing habits, the optimal hyperparameters of their DNNs

for next-word prediction may vary significantly. In addition, same as FL, there are also other important challenges in FBO, such as communication efficiency (i.e., the total number of exchanged parameters), rigorous privacy guarantees for the agents, decentralized communication, fairness among different agents, etc. As we will discuss in detail in Section 1.2.3, the works of [14, 15] have tackled a number of these challenges (e.g., retaining the raw data, agent heterogeneity, communication efficiency and rigorous privacy guarantees), whereas the others are still important open problems for FBO.

1.2.3 Overview of Representative Existing Works on FBO

The work of [15] has introduced the *federated Thompson sampling* (FTS) algorithm, which is the first algorithm for FBO. The FTS algorithm [15] has mainly tackled three of the major challenges faced by FBO (Section 1.2.2).

The first challenge results from the requirement to retain (hence not transmit) the raw data, which is a unique challenge faced by FBO yet not FL. Specifically, the information about a BO task is contained in its GP surrogate model (Section 1.2.1). However, unlike DNNs in standard FL for supervised learning, GPs are *nonparametric* [60]. Therefore, a BO task has no parameters (except for the raw data of BO) that can represent the GP surrogate and thus be exchanged among agents, while the raw data of BO should never be transmitted [39]. To overcome this challenge, the work of [15] has exploited RFFs (Section 1.2.1) to approximate a GP, which naturally yields parameters that contain the information about the approximate GP surrogate and can thus be communicated among the agents without exchanging the raw data. The second challenge concerns the communication efficiency, for which [15] has adopted TS as the acquisition function (Section 1.2.1), which reduces the number of exchanged parameters while maintaining competitive performances. The third challenge is caused by the potential heterogeneity among different agents, which is an important practical consideration in FL since different agents can have highly distinct properties [48]. In FBO, heterogeneity arises because different agents may have disparate objective functions. To address this challenge, [15] has derived a theoretical convergence guarantee to ensure that their FTS algorithm is asymptotically no-regret even when all agents have highly different objective functions.

More recently, the work of [14] has extended the FTS algorithm [15] by additionally addressing the challenge of rigorous privacy guarantees (Section 1.2.2) through the classic method of *differential privacy* (DP) [24]. Specifically, [14] has incorporated the general DP framework adopted by the DP-SGD [1] and DP-FedAvg [55] algorithms into (a modified version of) the FTS algorithm [15], to protect the *user-level privacy* of the agents in FBO. By achieving a guarantee on the user-level privacy, [14] guarantees that an adversary, even with arbitrary side information, cannot infer whether an agent has participated in FBO, hence

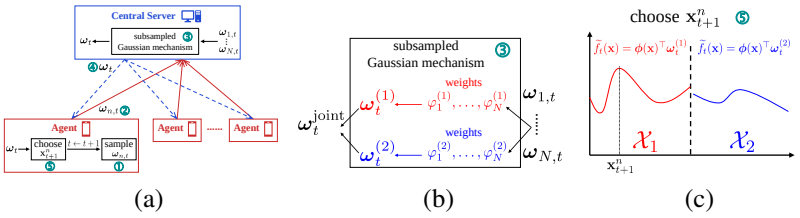


FIGURE 1.1 (a) DP-FTS algorithm without distributed exploration; (b)-(c) replacing steps (3) and (5) in (a) with that in (b) and (c) to derive the DP-FTS-DE algorithm ($P = 2$ sub-regions).

assuring every agent that its participation will not reveal its sensitive information. In addition, to compensate for the performance loss due to privacy preservation, [14] has proposed the technique of *distributed exploration* (DE) to further improve the practical performance (i.e., utility) of their algorithm. Intuitively, the DE technique is based on the idea of local modelling in GP [25] and is incorporated into the general DP framework in a seamless way thanks to the flexibility of the framework to model multiple parameter vectors [53].

Since the method introduced by [14], named DP-FTS-DE, is the state-of-the-art algorithm for FBO, we will introduce the details of the DP-FTS-DE algorithm in the next section (Section 1.2.4), and then discuss its theoretical guarantees and empirical performances in Section 1.2.5.

1.2.4 Algorithms for FBO

For ease of understanding, in this section, we will firstly introduce the DP-FTS algorithm without the DE technique, and then describe how DP-FTS can be modified to incorporate DE to derive the DP-FTS-DE algorithm.

The DP-FTS Algorithm. Fig. 1.1a illustrates the DP-FTS algorithm. Every iteration t of DP-FTS consists of the following steps:

Steps (1) and (2) by the agents: Every agent $\mathcal{A}_n$ samples a vector $\boldsymbol{\omega}_{n,t}$ following equation (1.2) using its own current history of t input-output pairs (step (1)), and then sends $\boldsymbol{\omega}_{n,t}$ to the central server – step (2).

Steps (3) and (4) by the central server: Next, the central server processes the N received vectors $\{\boldsymbol{\omega}_{n,t}\}_{n=1,\ldots,N}$ using a sub-sampled Gaussian mechanism – step (3) – which consists of four steps:

A. Selecting a random subset of agents $\mathcal{S}_t \subset \{1,\ldots,N\}$ by choosing each agent with probability q,

B. Clipping the vector $\boldsymbol{\omega}_{n,t}$ of every selected agent s.t. its L_2 norm is upper-bounded by S: $\widehat{\boldsymbol{\omega}}_{n,t} = \boldsymbol{\omega}_{n,t}/\max(1, \|\boldsymbol{\omega}_{n,t}\|_2 / S)$,

C. Calculating a weighted average of the clipped vectors with $\{\varphi_n\}_{n=1,\ldots,N}$: $\boldsymbol{\omega}_t = (q)^{-1} \sum_{n \in \mathcal{S}_t} \varphi_n \widehat{\boldsymbol{\omega}}_{n,t}$, and

D. Adding noise $\epsilon \sim \mathcal{N}(0, (zS\varphi_{\max}/q)^2)$ to $\boldsymbol{\omega}_t$ where $\varphi_{\max} = \max_{n=1,\ldots,N} \varphi_n$ and z is a parameter controlling the privacy-utility trade-off. The final vector $\boldsymbol{\omega}_t$ is then broadcast to all agents – step (4).

Step (5) by the agents: After an agent $\mathcal{A}_n$ receives the vector $\boldsymbol{\omega}_t$ from the central server, it can choose its next query $\mathbf{x}^n_{t+1}$ (step (5)). Choose $\delta \in (0, 1)$ and define $\beta_t := B + \sigma\sqrt{2(\gamma_{t-1} + 1 + \log(4/\delta)}$, in which B is an upper bound on the RKHS norm of the objective functions f^n's (Section 1.2.2), σ is the noise standard deviation (Section 1.2.1), and γ_{t-1} is the maximum information gain about f^n from any set of $t-1$ queries [70].

Then, with probability of $p_{t+1} \in (0, 1]$, $\mathcal{A}_n$ chooses $\mathbf{x}^n_{t+1}$ using standard TS by sampling a function f^n_{t+1} from its posterior of $\mathcal{GP}(\mu^n_t(\cdot), \beta^2_{t+1}\sigma^n_t(\cdot,\cdot)^2)$ (1.1) and then choosing $\mathbf{x}^n_{t+1} = \arg\max_{\mathbf{x} \in \mathcal{X}} f^n_{t+1}(\mathbf{x})$. Otherwise, with probability of $1 - p_{t+1}$, $\mathcal{A}_n$ chooses $\mathbf{x}^n_{t+1}$ using $\boldsymbol{\omega}_t$ received from the server: $\mathbf{x}^n_{t+1} = \arg\max_{\mathbf{x} \in \mathcal{X}} \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\omega}_t$. The sequence $(p_t)_{t \in \mathbb{Z}^+}$ is chosen as a monotonically increasing sequence such that $p_t \in (0, 1], \forall t$ and $p_t \to 1$ as $t \to \infty$. This ensures when $1 - p_t$ is large, an agent can leverage the information from the other agents (via $\boldsymbol{\omega}_t$) to improve its convergence by accelerating its exploration.

After choosing $\mathbf{x}^n_{t+1}$ and observing y^n_{t+1}, the agent $\mathcal{A}_n$ adds $(\mathbf{x}^n_{t+1}, y^n_{t+1})$ to its history of observations and then samples a new vector $\boldsymbol{\omega}_{n,t+1}$ – step (1). Next, $\mathcal{A}_n$ sends $\boldsymbol{\omega}_{n,t+1}$ to the central server – step (2)– and the algorithm is repeated.

The DP-FTS-DE Algorithm. The DP-FTS-DE algorithm is obtained by incorporating the technique of DE into DP-FTS described above.

Specifically, DE divides the entire domain $\mathcal{X}$ into $P \geq 1$ disjoint sub-regions $\mathcal{X}_1, \ldots, \mathcal{X}_P$, and the incorporation of DE requires three major modifications to DP-FTS. Firstly, during the initialization stage, every agent only explores a local sub-region $\mathcal{X}_i$ instead of the entire domain $\mathcal{X}$.

Next, instead of a single vector $\boldsymbol{\omega}_t$, the central server produces and broadcasts P vectors $\{\boldsymbol{\omega}^{(i)}_t\}_{i=1,\ldots,P}$, each corresponding to a different sub-region $\mathcal{X}_i$ and using a different set of weights $\{\varphi^{(i)}_n\}_{n=1,\ldots,N}$. Interestingly, the different privacy-preserving transformations performed by the central server to produce P vectors

can be interpreted as a single sub-sampled Gaussian mechanism producing a single joint vector $\boldsymbol{\omega}_t^{\text{joint}} \triangleq (\boldsymbol{\omega}_t^{(i)})_{i\in[P]}$ (Fig. 1.1b).

Then, if agent $\mathcal{A}_n$ uses the vector $\boldsymbol{\omega}_t^{\text{joint}} = (\boldsymbol{\omega}_t^{(i)})_{i\in[P]}$ received from the central server to choose the next query $\mathbf{x}_{t+1}^n$ (with probability $1 - p_{t+1}$), it chooses $\mathbf{x}_{t+1}^n$ by maximizing the reconstructed functions for all sub-regions, as illustrated in Fig. 1.1c. Refer to the work of [14] for a more complete description of the DP-FTS-DE algorithm.

1.2.5 Theoretical and Empirical Results for FBO

The work of [14] has provided theoretical guarantees on both the privacy and utility of DP-FTS-DE. The privacy guarantee is given by the proposition below:

Proposition 1. *There exist constants c_1 and c_2 such that for fixed q and T and any $\epsilon < c_1 q^2 T, \delta > 0$, DP-FTS-DE is (ϵ, δ)-DP if $z \geq c_2 q\sqrt{T\log(1/\delta)}/\epsilon$.*

The following theorem gives a guarantee on the utility/performance of DP-FTS-DE in terms an upper bound on R_T^1 (Section 1.2.2), in which all notations have been defined in Sections 1.2.1, 1.2.2 and 1.2.4:

Theorem 1 (Informal)**.** *With high probability,*

$$R_T^1 \quad = \quad \tilde{O}\Big(\big(B + 1/p_1\big)\, \gamma_T \sqrt{T} \;+\; \textstyle\sum_{t=1}^{T} \psi_t \;+\; B \sum_{t=1}^{T} \vartheta_t \Big).$$

$\psi_t := \tilde{O}((1-p_t) P \varphi_{\max} q^{-1} (\Delta_t + zS\sqrt{M}))$, *and* $\Delta_t := \sum_{n=1}^{N} \tilde{O}(\varepsilon B t^2 + B + \sqrt{M} + d_n + \sqrt{\gamma_t})$. $\vartheta_t := (1-p_t) \sum_{i=1}^{P} \sum_{n\in C_t} \varphi_n^{(i)}$, $C_t := \{n \in [N] \big| ||\boldsymbol{\omega}_{n,t}||_2 > S/\sqrt{P}\}$.

The first interesting insight from Theorem 1 is that agent $\mathcal{A}_1$ is asymptotically *no-regret* even if all other agents are heterogeneous, i.e., all other agents have significantly different objective functions from $\mathcal{A}_1$. This ensures that DP-FTS-DE is robust against the heterogeneity of agents, which is a significant challenge in FBO (Section 1.2.2).

Moreover, Theorem 1, when interpreted together with Proposition 1, also reveals some interesting theoretical insights regarding the **privacy-utility trade-off**. Firstly, a larger z (i.e., larger variance for the added Gaussian noise, Section 1.2.4) improves the privacy guarantee (Proposition 1) yet results in a worse utility since it leads to a worse regret upper bound (through ψ_t). Secondly, a larger q (i.e., more selected agents in each iteration, Section 1.2.4) improves the utility since it tightens the regret upper bound (by reducing the value of ψ_t) at the expense of a worse privacy guarantee (Proposition 1). The value of the clipping threshold S affects the regret upper bound (hence the utility) through two conflicting effects: a smaller S (1) reduces the value of ψ_t (hence, the regret bound) but is also likely to enlarge the cardinality of the set C_t which increases

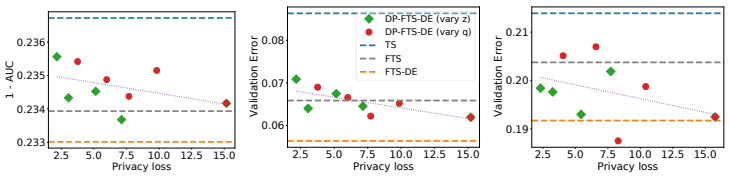


FIGURE 1.2 Scatter plots showing the trade-off between privacy loss and performance after 60 iterations for (a) the landmine detection, (b) human activity recognition, and (c) EMNIST experiments. In the above plots, plotted points leaning more towards the *left* (*bottom*) correspond to trade-off that favors privacy (utility), respectively.

the value of ϑ_t and hence, the regret bound (Theorem 1). Refer to Section 4 of [14] for a more detailed discussion on the theoretical guarantees of DP-FTS-DE, as well as the associated insights.

Empirical Results. The work of [14] has also demonstrated the practical effectiveness of DP-FTS-DE using three real-world experiments on landmine detection, human activity recognition, and EMNIST image classification. Some of the experimental results are displayed in Fig. 1.2, which show that FTS (from the work of [15]) consistently outperforms standard (non-federated) TS whereas FTS-DE (with the DE technique) further improves the performance of FTS. Moreover, Fig. 1.2 shows that with small privacy losses (in the single digit range [1]), DP-FTS-DE achieves a competitive performance (utility) and significantly outperforms standard TS in all settings. Furthermore, the figure also reveals a clear privacy-utility trade-off, i.e., a smaller privacy loss (more to the left) generally results in a worse utility (larger vertical value). Refer to Section 5 of the work of [14] for more details on the experimental settings, as well as more experimental results and discussions.

1.3 FEDERATED REINFORCEMENT LEARNING

In this section, we will firstly present some technical background on RL (Section 1.3.1) and FRL (Section 1.3.2). Next, we will briefly overview the representative existing works on FRL (Section 1.3.3) and discuss its framework (Section 1.3.4), as well as its corresponding theoretical and empirical results (Section 1.3.5).

1.3.1 Background on Reinforcement Learning

Reinforcement learning (RL) considers a discrete-time Markov decision process (MDP) [71]: $M \triangleq \{\mathcal{S}, \mathcal{U}, \mathcal{P}, \mathcal{R}, \gamma, \rho\}$, in which $\mathcal{S}$ and $\mathcal{U}$ represent the state space and action space respectively, $\mathcal{P}(s'|s,a)$ defines the transition probability from state s to s' after taking action a, $\mathcal{R}(s,a) : \mathcal{S} \times \mathcal{U} \to \mathbb{R}$ is the reward function for each state-action pair (s,a), $\gamma \in (0,1)$ is the discount

factor, and ρ is the initial state distribution. The behavior of an agent is controlled by a policy π, where $\pi(a|s)$ defines the probability that the agent chooses action a at state s. We consider episodic MDPs with trajectory horizon H. A trajectory $\tau \triangleq \{s_0, a_0, s_1, a_1, ..., s_{H-1}, a_{H-1}\}$ is a sequence of state-action pairs traversed by an agent following any stationary policy, where $s_0 \sim \rho$. $\mathcal{R}(\tau) \triangleq \sum_{t=0}^{H-1} \gamma^t \mathcal{R}(s_t, a_t)$ gives the discounted cumulative reward along τ.

Policy Gradient (PG) methods have achieved impressive successes in model-free RL [63, 64, etc.]. Compared with deterministic value function-based methods such as Q-learning [56], PG methods are generally more effective in high-dimensional problems and enjoy the flexibility of stochasticity. In PG, we use $\pi_{\boldsymbol{\theta}}$ to denote the policy parameterized by $\boldsymbol{\theta} \in \mathbb{R}^d$ (e.g., a neural network), and $p(\tau|\pi_{\boldsymbol{\theta}})$ to represent the trajectory distribution induced by the policy $\pi_{\boldsymbol{\theta}}$. For brevity, we use $\boldsymbol{\theta}$ to denote the corresponding policy $\pi_{\boldsymbol{\theta}}$. The performance of a policy $\boldsymbol{\theta}$ can be measured by $J(\boldsymbol{\theta}) \triangleq \mathbb{E}_{\tau \sim p(\cdot|\boldsymbol{\theta})}[\mathcal{R}(\tau)|M]$. Taking the gradient of $J(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ gives

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int_{\tau} \mathcal{R}(\tau) \nabla_{\boldsymbol{\theta}} p(\tau \mid \boldsymbol{\theta}) \mathrm{d}\tau = \mathbb{E}_{\tau \sim p(\cdot|\boldsymbol{\theta})} \left[\nabla_{\boldsymbol{\theta}} \log p(\tau \mid \boldsymbol{\theta}) \mathcal{R}(\tau) \mid M \right] \quad (1.3)$$

Then, the policy $\boldsymbol{\theta}$ can be optimized using gradient ascent. Since computing (1.3) is usually prohibitive, stochastic gradient ascent is typically used. In each iteration, we sample a batch of trajectories $\{\tau_i\}_{i=1}^B$ using the current policy $\boldsymbol{\theta}$, and update the policy by $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \widehat{\nabla}_B J(\boldsymbol{\theta})$, where η is the step size and $\widehat{\nabla}_B J(\boldsymbol{\theta})$ is an estimate of (1.3) using the sampled trajectories $\{\tau_i\}_{i=1}^B$: $\widehat{\nabla}_B J(\boldsymbol{\theta}) = \frac{1}{B} \sum_{i=1}^B \nabla_{\boldsymbol{\theta}} \log p\left(\tau_i \mid \boldsymbol{\theta}\right) \mathcal{R}\left(\tau_i\right)$. The commonly used policy gradient estimator, namely GPOMDP [6], can be expressed as

$$\begin{aligned} \widehat{\nabla}_B J(\boldsymbol{\theta}) &= \frac{1}{B} \sum_{i=1}^{B} g(\tau_i|\boldsymbol{\theta}), \\ g(\tau_i|\boldsymbol{\theta}) &= \sum_{h=0}^{H-1} \left[\sum_{t=0}^{h} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}\Big(a_t \mid s_t\Big) \right] \Big(\gamma^h r(s_h, a_h) - C_{b_h}\Big), \end{aligned} \quad (1.4)$$

in which $\tau_i = \{s_0^i, a_0^i, s_1^i, a_1^i, \ldots, s_{H-1}^i, a_{H-1}^i\}$ and $g(\tau_i|\boldsymbol{\theta})$ is an *unbiased* estimation of $\nabla_{\boldsymbol{\theta}} \log p(\tau_i \mid \boldsymbol{\theta}) \mathcal{R}(\tau_i)$ [58, 77].

To improve the sample efficiency of the estimation of (1.4), *stochastic variance-reduced gradient* (SVRG) [4, 34, 61, 75] has recently been adopted to reduce the variance of policy gradient estimators. The work of [58] has adapted the theoretical analysis of SVRG to PG to introduce the *stochastic variance-reduced policy gradient* (SVRPG) algorithm. More recently, [77] has refined the analysis of SVRPG [58] and shown that SVRPG enjoys a sample complexity of $O(1/\epsilon^{5/3})$.

1.3.2 Background on Federated Reinforcement Learning

It is well established that RL systems suffer from poor sample efficiency in real-world applications [23, 42], which has motivated the development of *federated RL* (FRL) [86]. Here, we present the problem statement of FRL following the work of [26]. In FRL, K distributed agents/computing nodes $\mathcal{A}^{(1)}, \ldots, \mathcal{A}^{(K)}$ are operating in separate copies of the same underlying MDP with random initial state distributions $\rho^{(k)}$'s. Each agent independently runs an RL algorithm by interacting with its environment. Like FBO, FRL involves a central server which is assumed to be trustworthy and governs the federation process. The objective/incentive for agents to participate in the federation is to obtain a well-performing policy $\boldsymbol{\theta}^*$ with *fewer* agent-environment interactions compared to independent learning on their own. As a representative motivating example, K hospitals, each of which only possesses a limited number of admission records, who apply RL algorithms to provide clinical decision support may wish to collaboratively discover better treatment protocols for the same disease. To improve the sample efficiency of this application, in every iteration, each agent (hospital) $\mathcal{A}^{(k)}$ interacts with its environment following a parameterized policy (clinical trial protocols) $\boldsymbol{\theta}^{(k)}$ and then shares some information with the trustworthy server (e.g., government), who aggregates all received information and then broadcasts it back to all agents to improve their subsequent decision making.

FRL assumes that all participating RL agents share the same sets of states $\mathcal{S}$ and actions $\mathcal{U}$ as a common ground for collaboration. In addition, an FRL system is concerned with only a single task (e.g., treatment for a single disease), and hence the underlying MDP M (i.e., the transition probability $\mathcal{P}$ and reward function $\mathcal{R}$) is assumed to be the same for all agents. Similar to RL discussed above, every agent $\mathcal{A}^{(k)}$ in FRL aims to maximize the objective function $J^{(k)} \triangleq \mathbb{E}_{\tau \sim p(\cdot|\boldsymbol{\theta}^{(k)})}[\mathcal{R}(\tau)|M]$ with *fewer* interactions with the environment compared with independent learning. That is, to reach a certain performance threshold, the required total number of trajectories $|\tau^{(k)}|$ of agent-environment interactions of agent $\mathcal{A}^{(k)}$ should be smaller than that of independent learning on its own. In other words, the agent aims to improve its sample efficiency by joining the federation. To ensure that FRL leads to a speedup in sample efficiency with a larger number of agents, [26] has presented a convergence guarantee for their proposed federated policy gradient framework and derived an upper bound on the average number of trajectories required by each agent for the policy to reach an ϵ-stationary performance point (Section 1.3.5). In their theoretical analysis, the objective function $J(\boldsymbol{\theta})$ is assumed to be L-smooth, which is a commonly adopted assumption in non-convex optimization [4, 61] and recent convergence results of policy gradient [58, 77]. Furthermore, the initial states of the agents are assumed to be uniformly distributed, i.e., the $\rho^{(k)}$'s are uniform distributions.

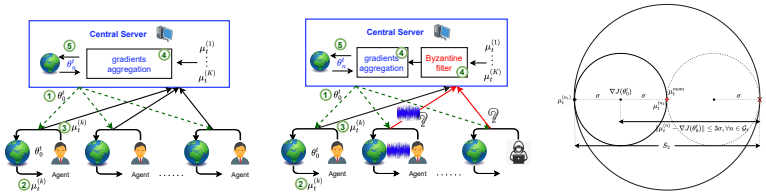
Core Principles and Challenges of FRL. In addition to the fundamental

challenge of retaining the raw data (i.e., the trajectories of agent-environment interactions $\tau^{(k)}$ shall never be shared), FRL is faced with a few other critical challenges. Firstly, with N agents independently interacting with their own copy of the environment, does FRL theoretically guarantee the convergence and proportionally improved sampled efficiency thanks to the federation? Unlike FL where the training data can be collected offline, FRL requires every agent to sample trajectories online by interacting with the environment, which can be slow and expensive. Therefore, theoretical guarantees on the convergence and sample efficiency speedup provide assurance and incentives for practical applications. Secondly, FRL systems need to account for fault tolerance, which is a critical challenge in practical decision-making applications since distributed systems are likely to fail periodically. Being fault-tolerant allows the FRL system to continue operating properly without interruption when one or more of its agents fail. Furthermore, the heterogeneity among different agents (e.g., the difference in their initial state distributions, computational budgets, etc.) is another crucial challenge for FRL. For example, since different hospitals may administer patients from distinct geographic locations, their initial state distributions may vary significantly. Similarly, hospitals may possess different computational budgets and hence can adopt disparate policy parameterization schemes and optimization methods. Moreover, FRL is also faced with other challenges which plague FL in general, such as rigorous privacy guarantees, decentralized communication, fairness, etc. The representative work of [26], which we will discuss in Section 1.3.3, has tackled a number of these challenges (e.g., retaining the raw data, guaranteed convergence and proportional sample efficiency speedup, and fault tolerance), whereas the others remain important open problems for FRL.

1.3.3 Overview of Representative Existing Works on FRL

Firstly introduced by [86], FRL has been applied to a number of practical applications [50, 51, 57, etc.]. The first general-purpose framework for FRL is the *federated Policy Gradient with Byzantine Resilience* (FedPG-BR) framework introduced by [26]. The FedPG-BR framework [26] has mainly addressed three of the aforementioned challenges faced by FRL (Section 1.3.2). For brevity, we defer the details of the FedPG-BR algorithm to Section 1.3.4, followed by discussions on its theoretical guarantees and empirical performances in Section 1.3.5.

The first challenge addressed by [26] is regarding the theoretical guarantees on the convergence and sample efficiency improvements. To this end, [26] has adopted the *stochastically controlled stochastic gradient* (SCSG) algorithm [41], which is a variant of SVRG (Section 1.3.1), in the federated policy gradient framework, which provides a refined control over the variance of the gradient estimation. Their theoretical analysis has shown that their proposed framework is guaranteed to converge and enjoys a proportional sample efficiency speedup with respect to



(a) (b) (c)

FIGURE 1.3 Workflows of (a) the Federated Policy Gradient algorithm (without fault tolerance) and (b) the Federated Policy Gradient algorithm with Byzantine Resilience Filter (i.e., the FedPG-BR framework); and (c) visualizing diagram of the concentration bound constructed by the BR filter.

the number of agents. The second challenge results from the fact that distributed systems are vulnerable to random failures or adversarial attacks coming from the distributed agents, which may slow down or completely disrupt the convergence of the FRL systems. These random failures and adversarial attacks are modeled as *Byzantine faults*, which is the most stringent fault formalism in distributed computing [7, 52]. Byzantine faults consider a distributed computing system with up to half of the computing nodes being random failures or adversarial attackers. They are hard to detect when the raw data of the agents is not accessible to the server, which makes them a considerable practical challenge in FL [79]. To address this challenge, [26] has designed a gradient-based Byzantine filter on top of their proposed federated policy gradient framework, in order to remove those gradients that are likely sent by Byzantine agents with high probability. This has enabled the algorithm of [26] to be tolerant to Byzantine faults when less than half of the agents are Byzantine agents.

1.3.4 Frameworks and Algorithms for FRL

In this section, we will describe the details of the FedPG-BR framework [26], which consists of the *Federated Policy Gradient* (FedPG) algorithm and the *Byzantine Resilience* (BR) filter.

The FedPG Algorithm. Fig. 1.3a illustrates the FedPG algorithm. Every iteration t of FedPG includes the following steps:

Step (1) by the central server: FedPG starts with a randomly initialized parameter $\tilde{\boldsymbol{\theta}}_0$ at the server. At the beginning of the t-th iteration, the server keeps a snapshot of its parameter from the previous iteration (i.e., $\boldsymbol{\theta}_0^t \leftarrow \tilde{\boldsymbol{\theta}}_{t-1}$) and broadcasts this parameter to all agents.

Step (2) and (3) by the agents: Every agent $\mathcal{A}^{(k)}$ samples B_t trajectories $\{\tau_{t,i}^{(k)}\}_{i=1}^{B_t}$ using the policy $\boldsymbol{\theta}_0^t$, computes a gradient estimate $\mu_t^{(k)} \triangleq$

$1/B_t \sum_{i=1}^{B_t} g(\tau_{t,i}^{(k)}|\boldsymbol{\theta}_0^t)$ following Equation (1.4) – step (2) – and sends $\mu_t^{(k)}$ back to the server – step (3).

Steps (4) and (5) by the central server: The server computes the batch gradient μ_t by averaging the received gradient estimates – step (4) – and updates the policy parameters $\boldsymbol{\theta}_0^t$ via the following semi-stochastic gradient –step (5) – according to SCSG optimization [41]: $v_n^t \triangleq \frac{1}{b_t} \sum_{j=1}^{b_t} \left[g(\tau_{n,j}^t|\boldsymbol{\theta}_n^t) - g(\tau_{n,j}^t|\boldsymbol{\theta}_0^t) \right] + \mu_t$, in which b_t trajectories are sampled *independently* by the server on a separate copy of the MDP M and the number N_t of optimization steps is sampled from a geometric distribution with the parameter $\frac{B_t}{B_t+b_t}$. The server then stores a new snapshot of the updated parameter, and the algorithm is repeated.

The BR Filter. Fig. 1.3b demonstrates the complete FedPG-BR framework, which is obtained by incorporating the Byzantine Resilience filter (Fig. 1.3c) into the FedPG algorithm described above. After receiving all the gradients, the server applies the BR filter (step ④), which is designed based on the following assumptions: firstly, at any given iteration t, there are less than $K/2$ Byzantine agents; secondly, the gradient estimates sent by the good (non-Byzantine) agents should concentrate in a region containing the true analytical gradient defined in (1.3). In particular, the work of [26] has assumed that there exists a constant σ such that $\|g(\tau|\boldsymbol{\theta}) - \nabla J(\boldsymbol{\theta})\| \leq \sigma$ for any $\tau \sim p(\cdot|\boldsymbol{\theta})$ and any policy $\pi_{\boldsymbol{\theta}}$, which implies that the discrepancy between any two gradient estimates is at most 2σ in the Euclidean space. Therefore, the BR filter computes the pair-wise Euclidean distances of all gradients[2] and selects a set of gradients $S_2 \triangleq \{\mu_t^{(k)}\}$ that are close (within 2σ in Euclidean distance) to more than $K/2$ other gradients. Any gradient $\mu_t^{(\tilde{k})}$ that is the closest to the mean of S_2 will be noted as the Mean of Median vector of all received gradients, denoted as μ_t^{mom}. Thereafter, the BR filter removes any received gradient $\mu_t^{(k')}$ whose distance to μ_t^{mom} is more than 2σ from the gradients aggregation step (step step ④). Essentially, the BR filter constructs a concentration bound such that the server aggregates only those gradient estimates that are not far away from the true analytical gradient. Refer to the work of [26] for a more complete description of the FedPG-BR framework.

1.3.5 Theoretical and Empirical Results for FRL

The work of [26] has provided theoretical guarantees on the convergence of FedPG-BR:

[2] It can be implemented using the Euclidean Distance Matrix Trick [3].

TABLE 1.1 Sample complexities of relevant works to achieve $\mathbb{E}\|\nabla J(\boldsymbol{\theta})\|^2 \leq \epsilon$.

SETTINGS	METHODS	COMPLEXITY
$K = 1$	REINFORCE [74]	$O(1/\epsilon^2)$
	GPOMDP [6]	$O(1/\epsilon^2)$
	SVRPG [58]	$O(1/\epsilon^2)$
	SVRPG [77]	$O(1/\epsilon^{5/3})$
	FedPG-BR	$O(1/\epsilon^{5/3})$
$K > 1, \alpha = 0$	**FedPG-BR**	$O(\frac{1}{\epsilon^{5/3}K^{2/3}})$
$K > 1, \alpha > 0$	**FedPG-BR**	$O(\frac{1}{\epsilon^{5/3}K^{2/3}} + \frac{\alpha^{4/3}}{\epsilon^{5/3}})$

Theorem 2 (Informal)**.** *The output of FedPG-BR, $\tilde{\boldsymbol{\theta}}$, satisfies:*

$$\mathbb{E}[\|\nabla J(\tilde{\boldsymbol{\theta}}_a)\|^2] \;\leq\; \frac{2\Psi\left[J(\tilde{\boldsymbol{\theta}}^*) - J(\tilde{\boldsymbol{\theta}}_0)\right]}{TB^{1/3}} \;+\; \frac{8\sigma^2}{(1-\alpha)^2 KB} \;+\; \frac{96\alpha^2\sigma^2 V}{(1-\alpha)^2 B}$$

where α denotes the ratio of Byzantine agents s.t. $0 \leq \alpha < 0.5$ and $\tilde{\boldsymbol{\theta}}^*$ is a global maximizer of J. Ψ is a smoothness constant and V is another constant used to refine the concentration bound of the BR filter. For brevity, readers are referred to [26] for the detailed definitions of Ψ and V.

This theorem leads to many interesting insights, such as the sample complexity (of each agent in FedPG-BR) to reach an ϵ-stationary point which is summarized in Table 1.1. The table reveals that the sample complexity of FedPG-BR in the single-agent setting matches that of SVRPG derived by [77]. In addition, when $K > 1$, $\alpha = 0$, Table 1.1 shows that the total number of trajectories required by each agent is upper-bounded by $O(1/(\epsilon^{5/3}K^{2/3}))$. This result gives us the theoretical grounds to encourage more agents to participate in the federation since the number of trajectories each agent needs to sample decays at a rate of $O(1/K^{2/3})$. Furthermore, for a more realistic system where an α-fraction ($\alpha > 0$) of the agents are Byzantine agents, the results in Table 1.1 assure us that the total number of trajectories required by each agent will be increased by *only an additive term of* $O(\alpha^{4/3}/\epsilon^{5/3})$, the impact of which vanishes when $\alpha \to 0$.

Empirial Results. The work of [26] has also demonstrated the practical effectiveness of FedPG-BR using standard RL benchmarks including CartPole balancing [5], LunarLander, and the 3D continuous locomotion control task of HalfCheetah [21]. The experimental results shown in Fig. 1.4 suggest that the performance of FedPG-BR is comparable to SVRPG in the single-agent training setting ($K = 1$), and is improved significantly with the federation of $K = 3$ and $K = 10$ agents. This corroborates our theoretical insights implying that the sample efficiency of FedPG-BR is guaranteed to improve proportionally with

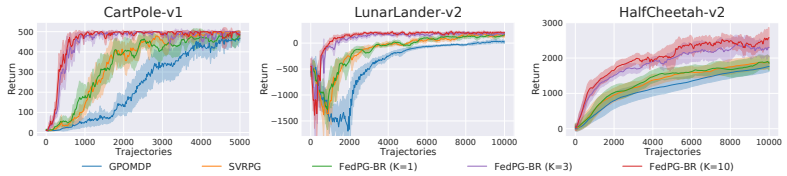


FIGURE 1.4 Performance of FedPG-BR in ideal systems with $\alpha = 0$ for the three tasks.

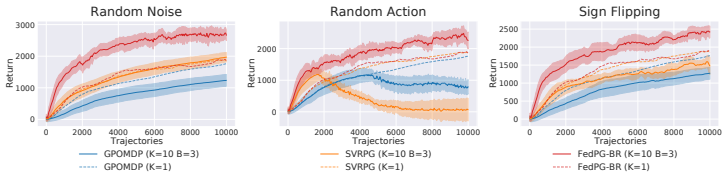


FIGURE 1.5 Performance of FedPG-BR in practical systems with $\alpha > 0$ for HalfCheetah. Each subplot corresponds to a different type of Byzantine failure exercised by the 3 Byzantine agents.

the number of agents K. Furthermore, Fig. 1.5 shows another experiment on the HalfCheetah task with $K = 10$ agents, among which 3 are Byzantine agents.

The Byzantine agents are simulated using the following three schemes: *Random Noise (RN)*: each Byzantine agent sends a random vector to the server; (b) *Random Action (RA)*: every Byzantine agent ignores the policy from the server and takes actions randomly, which is used to simulate random system failures (e.g., hardware failures); (c) *Sign Flipping (SF)*: each Byzantine agent computes the correct gradient but sends the scaled negative gradient, which is used to simulate adversarial attacks aiming to manipulate the direction of policy updates at the server. The three subplots show that the existence of only 3 Byzantine agents causes the performance of federated policy gradient systems (GPOMDP and SVRPG) to be worse than that in the single-agent setting. In contrast, our FedPG-BR is robust against all three types of Byzantine failures. That is, FedPG-BR ($K = 10$ $B = 3$) with 3 Byzantine agents still significantly outperforms the single-agent setting, and more importantly, *performs comparably to FedPG-BR ($K = 10$) with 10 good agents* (shown in the rightmost subplot in Fig. 1.4). These results demonstrate that even in practical systems with random failures or adversarial attacks, FedPG-BR is still able to deliver superior performances, providing an assurance on the reliability of the framework to promote its practical deployment and significantly improving the practicality of FRL.

1.4 RELATED WORK

1.4.1 Federated Bayesian Optimization

To the best of our knowledge, the works of [14, 15] (Section 1.2.3) are the only works on FBO to date. However, a number of previous works on BO have considered similar settings to the FBO setting, which we review here.

The work of [68] has also considered collaborative BO involving multiple agents. However, since their method is not designed for the FBO setting, they did not address the important challenges faced by FBO (e.g., they have assumed that all agents share the same objective function and hence did not consider the issue of heterogeneity), but have instead focused on the issue of fairness among the agents [68]. Furthermore, a number of works such as [13, 65] have studied BO in the multi-agent setting from the perspective of *game theory*. FBO also shares similarities with *batch/parallel BO*, which has seen a large body of works in recent years [12, 18, 20, 27, 30, 37, 72]. However, batch BO has many important differences with FBO. For example, batch BO aims to maximize a single objective function whereas FBO allows different agents to have different objective functions, and the sharing of the raw observations in batch BO is not prohibited. Another popular line of works on BO which is also related to FBO is *multi-fidelity BO* [17, 36, 83, 84]. This is because, from the point of view of an agent in FBO, the information from the other agents (received via the central server) can be considered as low-fidelity information.

1.4.2 Federated Reinforcement Learning

FRL was initially introduced by [86] which studied how to collaboratively learn two Q-networks when one of the two agents does not receive rewards from the MDP. Thereafter, FRL has been applied to a number of practical applications, such as autonomous driving [50, 62], fast personalization [57], robots navigation [51], traffic sign control [80], and resource management in networking [82]. Despite their promising applications, these works were not equipped with theoretical guarantees and hence did not provide theoretical understandings to the important challenges faced by FRL we have discussed in Section 1.3.2.

The work of [26] has presented the first federated policy gradient framework in the FRL setting which guarantees a proportional sample efficiency improvement with respect to the number of agents and achieves robustness against faulty (Byzantine) agents, hence tackling the challenges of theoretical guarantees and fault tolerance (Section 1.3.2). Regarding the challenge of the heterogeneity among different agents, the work of [76] has defined the heterogeneity in FRL as the data heterogeneity resulting from the difference in the initial state distributions and environment dynamics among different agents. As a result, [76] has proposed an FRL framework to tackle such heterogeneity in FRL by penalizing

the Kullback-Leibler (KL) divergence (i.e., encouraging the similarity) between the local policies and the global policy.

FRL also bears close similarity to another line of works on distributed and parallel RL [2, 8, 31, 38], because some of the challenges faced by FRL (Section 1.3.2), such as theoretical guarantees and fault tolerance, are also important considerations for these works. However, there are also substantial differences between these settings and FRL, since FRL also poses some unique challenges such as the requirement to retain the raw data. One representative work in distributed and parallel RL is [9], which has proposed the Byzantine-robust mean estimation technique for both online and offline distributed RL.

1.4.3 Federated Bandits

Multi-armed bandits is another popular sequential decision-making problem [40] with important applications such as recommender systems. A growing number of recent works have introduced federated versions of classic bandit algorithms.

Some works have focused on federated K-armed bandits where the number K of arms is finite and the arms are not associated with feature vectors. The works of [46] and [47] have focused on federated K-armed bandits with rigorous privacy guarantees, and have considered both centralized and decentralized communication. The work of [66] has proposed a setting for federated K-armed bandits in which the goal is to minimize the regrets of a global bandit model, which has the same set of arms as the agents. The reward of every arm for the global bandit model is the average of the rewards of the corresponding arm from all agents. The more recent work of [67] has extended [66] to incorporate personalization such that every agent attempts to maximize a weighted combination between the global and its local rewards. After that, subsequent works have focused on other aspects of federated K-armed bandits, such as decentralized communication via the gossiping algorithm [85], the security of federated K-armed bandits through cryptographic techniques [11], uncoordinated exploration [78], and robustness against Byzantine attacks [19].

In addition, a number of works have considered the problem of federated linear contextual bandits. The work of [73] has proposed a distributed linear contextual bandit algorithm that allows every agent to use the observations from the other agents without requiring them to share their raw data. Subsequently, the work of [22] has extended the method from [73] to incorporate differential privacy and decentralized communication. The work of [32] has considered a setting of federated linear contextual bandits where every agent is associated with a unique context vector. The works of [29, 43] have both focused on extending the method from [73] in order to allow asynchronous communication. The work of [33] has focused on the robustness of federated linear contextual bandits against

Byzantine attacks. A few recent works have relaxed the assumption of a linear reward function adopted by linear contextual bandits to consider non-linear reward functions. The work of [16] has introduced the first *federated neural bandit* algorithm which is able to exploit neural networks to learn the reward function. Moreover, the works of [49] and [69] have, respectively, extended $\chi-$armed bandits and combinatorial bandits to the federated setting.

1.5 OPEN PROBLEMS AND FUTURE DIRECTIONS

As we have discussed in this chapter, extending classic sequential decision-making algorithms (e.g., BO and RL) into the federated setting holds enormous potential yet is also faced with immense challenges. Some of these challenges have been addressed by recent works, such as the works of [14, 15] for FBO (Section 1.2.3) and the work of [26] for FRL (Section 1.3.3). However, some other challenges remain to be tackled and are hence still important open problems.

For FBO, note that the theoretical analyses from the works of [14, 15] have focused on the robustness against agent heterogeneity. Therefore, it remains an open problem to theoretically show the benefit of the federation when the agents are in fact homogeneous (or when the degree of heterogeneity is low), i.e., to show that a larger number of agents lead to better regret upper bounds. Furthermore, extending FBO algorithms to cater to other important considerations of the federated setting also represents important future directions in order to make FBO algorithms more practical. For example, allowing decentralized and asynchronous communication in FBO is an important challenge and a meaningful extension, since it allows FBO to work under more flexible communication protocols. Ensuring fairness among different agents in FBO is another significant open problem, since societal issues such as fairness have been receiving growing concerns and are hence crucial considerations for the real-world deployment of FBO algorithms. To this end, the method from [68], which has studied fairness in collaborative BO among multiple agents, may provide interesting inspirations.

For FRL, although the work of [26] has managed to avoid the transmission of the raw data (i.e., RL trajectories) of the agents, they have not provided a rigorous privacy guarantee. Therefore, it is an intriguing future direction to equip FRL systems with rigorous privacy guarantees (e.g., via differential privacy) in order to protect the privacy of the participating agents in a principled way. In addition, the theoretical analysis of [26] has focused on improving the sample efficiency for homogeneous agents. Therefore, a natural future extension is to derive theoretical guarantees when the agents are in fact heterogeneous and to study situations under which the benefits of the federation can be guaranteed in the presence of heterogeneity. Furthermore, designing decentralized FRL systems and achieving fairness among different FRL agents also represent promising future directions for FRL.

1.6 ACKNOWLEDGEMENT

This research is part of the programme DesCartes and is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

BIBLIOGRAPHY

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] Mridul Agarwal, Bhargav Ganguly, and Vaneet Aggarwal. Communication efficient parallel reinforcement learning. In *Uncertainty in Artificial Intelligence*, pages 247–256. PMLR, 2021.

[3] Samuel Albanie. Euclidean distance matrix trick. Technical report, 2019.

[4] Zeyuan Allen-Zhu and Elad Hazan. Variance reduction for faster non-convex optimization. In *International conference on machine learning*, pages 699–707, 2016.

[5] Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

[6] Jonathan Baxter and Peter L Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

[7] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186, 1999.

[8] Tianyi Chen, Kaiqing Zhang, Georgios B Giannakis, and Tamer Başar. Communication-efficient policy gradient methods for distributed reinforcement learning. *IEEE Transactions on Control of Network Systems*, 9(2):917–929, 2021.

[9] Yiding Chen, Xuezhou Zhang, Kaiqing Zhang, Mengdi Wang, and Xiaojin Zhu. Byzantine-robust online and offline distributed reinforcement learning. arXiv:2206.00165, 2022.

[10] Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. In *Proc. ICML*, pages 844–853, 2017.

[11] Radu Ciucanu, Pascal Lafourcade, Gael Marcadet, and Marta Soare. Samba: A generic framework for secure federated multi-armed bandits. *Journal of Artificial Intelligence Research (JAIR)*, 2022.

[12] Emile Contal, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis. Parallel Gaussian process optimization with upper confidence bound and pure exploration. In *Proc. ECML/PKDD*, pages 225–240, 2013.

[13] Zhongxiang Dai, Yizhou Chen, Bryan Kian Hsiang Low, Patrick Jaillet, and Teck-Hua Ho. R2-B2: Recursive reasoning-based Bayesian optimization for no-regret learning in games. In *Proc. ICML*, 2020.

[14] Zhongxiang Dai, Bryan Kian Hsiang Low, and Patrick Jaillet. Differentially private federated Bayesian optimization with distributed exploration. In *Proc. NeurIPS*, volume 34, 2021.

[15] Zhongxiang Dai, Kian Hsiang Low, and Patrick Jaillet. Federated Bayesian optimization via Thompson sampling. In *Proc. NeurIPS*, 2020.

[16] Zhongxiang Dai, Yao Shu, Arun Verma, Flint Xiaofeng Fan, Bryan Kian Hsiang Low, and Patrick Jaillet. Federated neural bandits. In *Proc. ICLR*, 2023.

[17] Zhongxiang Dai, Haibin Yu, Bryan Kian Hsiang Low, and Patrick Jaillet. Bayesian optimization meets Bayesian optimal stopping. In *Proc. ICML*, pages 1496–1506, 2019.

[18] Erik A Daxberger and Bryan Kian Hsiang Low. Distributed batch Gaussian process optimization. In *Proc. ICML*, pages 951–960, 2017.

[19] Ilker Demirel, Yigit Yildirim, and Cem Tekin. Federated multi-armed bandits under byzantine attacks. arXiv:2205.04134, 2022.

[20] Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization. *Journal of Machine Learning Research*, 15:3873–3923, 2014.

[21] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[22] Abhimanyu Dubey and AlexSandy' Pentland. Differentially-private federated linear bandits. In *Proc. NeurIPS*, volume 33, pages 6003–6014, 2020.

[23] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. arXiv:1904.12901, 2019.

[24] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[25] David Eriksson, Michael Pearce, Jacob Gardner, Ryan D. Turner, and Matthias Poloczek. Scalable global optimization via local Bayesian optimization. In *Proc. NeurIPS*, 2019.

[26] Flint Xiaofeng Fan, Yining Ma, Zhongxiang Dai, Wei Jing, Cheston Tan, and Bryan Kian Hsiang Low. Fault-tolerant federated reinforcement learning with theoretical guarantee. In *Proc. NeurIPS*, 2021.

[27] Javier Garcia-Barcos and Ruben Martinez-Cantin. Fully distributed Bayesian optimization with stochastic policies. In *Proc. IJCAI*, 2019.

[28] Roman Garnett. *Bayesian Optimization*. Cambridge Univ. Press, 2022.

[29] Jiafan He, Tianhao Wang, Yifei Min, and Quanquan Gu. A simple and provably efficient algorithm for asynchronous federated contextual linear bandits. arXiv:2207.03106, 2022.

[30] José Miguel Hernández-Lobato, James Requeima, Edward O Pyzer-Knapp, and Alán Aspuru-Guzik. Parallel and distributed Thompson sampling for large-scale accelerated exploration of chemical space. In *Proc. ICML*, 2017.

[31] Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado Van Hasselt, and David Silver. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018.

[32] Ruiquan Huang, Weiqiang Wu, Jing Yang, and Cong Shen. Federated linear contextual bandits. In *Proc. NeurIPS*, volume 34, 2021.

[33] Ali Jadbabaie, Haochuan Li, Jian Qian, and Yi Tian. Byzantine-robust federated linear bandits. arXiv:2204.01155, 2022.

[34] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323, 2013.

[35] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. arXiv:1912.04977, 2019.

[36] Kirthevasan Kandasamy, Gautam Dasarathy, Junier B Oliva, Jeff Schneider, and Barnabás Póczos. Gaussian process bandit optimisation with multi-fidelity evaluations. In *Proc. NeurIPS*, pages 992–1000, 2016.

[37] Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabás Póczos. Parallelised Bayesian optimisation via Thompson sampling. In *Proc. AISTATS*, pages 133–142, 2018.

[38] R Matthew Kretchmar. Parallel reinforcement learning. In *The 6th World Conference on*

Systemics, Cybernetics, and Informatics. Citeseer, 2002.

[39] Matt Kusner, Jacob Gardner, Roman Garnett, and Kilian Weinberger. Differentially private Bayesian optimization. In *Proc. ICML*, pages 918–927, 2015.

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

[41] Lihua Lei, Cheng Ju, Jianbo Chen, and Michael I Jordan. Non-convex finite-sum optimization via scsg methods. In *Advances in Neural Information Processing Systems*, pages 2348–2358, 2017.

[42] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv:2005.01643, 2020.

[43] Chuanhao Li and Hongning Wang. Asynchronous upper confidence bound algorithms for federated linear bandits. In *Proc. AISTATS*, 2022.

[44] Qinbin Li, Zeyi Wen, and Bingsheng He. Practical federated gradient boosting decision trees. In *Proc. AAAI*, pages 4642–4649, 2020.

[45] Qinbin Li, Zhaomin Wu, Zeyi Wen, and Bingsheng He. Privacy-preserving gradient boosting decision trees. In *Proc. AAAI*, pages 784–791, 2020.

[46] Tan Li and Linqi Song. Privacy-preserving communication-efficient federated multi-armed bandits. *IEEE Journal on Selected Areas in Communications*, 2022.

[47] Tan Li, Linqi Song, and Christina Fragouli. Federated recommendation system via differential privacy. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 2592–2597. IEEE, 2020.

[48] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. arXiv:1908.07873, 2019.

[49] Wenjie Li, Qifan Song, Jean Honorio, and Guang Lin. Federated x-armed bandit. arXiv:2205.15268, 2022.

[50] Xinle Liang, Yang Liu, Tianjian Chen, Ming Liu, and Qiang Yang. Federated transfer reinforcement learning for autonomous driving. arXiv:1910.06001, 2019.

[51] Boyi Liu, Lujia Wang, and Ming Liu. Lifelong federated reinforcement learning: a learning architecture for navigation in cloud robotic systems. *IEEE Robotics and Automation Letters*, 4(4):4555–4562, 2019.

[52] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

[53] H Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nicolas Papernot, and Peter Kairouz. A general approach to adding differential privacy to iterative training procedures. In *Proc. NeurIPS Workshop on Privacy Preserving Machine Learning*, 2018.

[54] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In *Proc. AISTATS*, 2017.

[55] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *Proc. ICLR*, 2018.

[56] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. arXiv:1312.5602, 2013.

[57] Chetan Nadiger, Anil Kumar, and Sherine Abdelhak. Federated reinforcement learning for fast personalization. In *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 123–127. IEEE, 2019.

[58] Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. Stochastic variance-reduced policy gradient. In *Proceedings of ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 4026–4035. PMLR, 10–15 Jul 2018.

[59] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Proc. NeurIPS*, pages 1177–1184, 2008.

[60] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[61] Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323, 2016.

[62] Thomas Rudolf, Tobias Schürmann, Matteo Skull, Stefan Schwab, and Sören Hohmann. Data-driven automotive development: Federated reinforcement learning for calibration and control. In *22. Internationales Stuttgarter Symposium*, pages 369–384. Springer, 2022.

[63] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

[64] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv:1707.06347, 2017.

[65] Pier Giuseppe Sessa, Ilija Bogunovic, Maryam Kamgarpour, and Andreas Krause. No-regret learning in unknown games with correlated payoffs. In *Proc. NeurIPS*, 2019.

[66] Chengshuai Shi and Cong Shen. Federated multi-armed bandits. In *Proc. AAAI*, 2021.

[67] Chengshuai Shi, Cong Shen, and Jing Yang. Federated multi-armed bandits with personalization. In *Proc. AISTATS*, pages 2917–2925. PMLR, 2021.

[68] Rachael Hwee Ling Sim, Yehong Zhang, Bryan Kian Hsiang Low, and Patrick Jaillet. Collaborative Bayesian optimization with fair regret. In *Proc. ICML*, pages 9691–9701. PMLR, 2021.

[69] Sambhav Solanki, Samhita Kanaparthy, Sankarshan Damle, and Sujit Gujar. Differentially private federated combinatorial bandits with constraints. arXiv:2206.13192, 2022.

[70] Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proc. ICML*, pages 1015–1022, 2010.

[71] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[72] Arun Verma, Zhongxiang Dai, and Bryan Kian Hsiang Low. Bayesian optimization under stochastic delayed feedback. In *Proc. ICML*, pages 22145–22167. PMLR, 2022.

[73] Yuanhao Wang, Jiachen Hu, Xiaoyu Chen, and Liwei Wang. Distributed bandit learning: Near-optimal regret with efficient communication. In *Proc. ICLR*, 2020.

[74] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[75] Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

[76] Zhijie Xie and SH Song. Fedkl: Tackling data heterogeneity in federated reinforcement learning by penalizing kl divergence. arXiv:2204.08125, 2022.

[77] Pan Xu, Felicia Gao, and Quanquan Gu. An improved convergence analysis of stochastic variance-reduced policy gradient. In *Uncertainty in Artificial Intelligence*, pages 541–551. PMLR, 2020.

[78] Zirui Yan, Quan Xiao, Tianyi Chen, and Ali Tajer. Federated multi-armed bandit via uncoordinated exploration. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5248–5252. IEEE, 2022.

[79] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3):1–207,

2019.

[80] Yutong Ye, Wupan Zhao, Tongquan Wei, Shiyan Hu, and Mingsong Chen. Fedlight: Federated reinforcement learning for autonomous multi-intersection traffic signal control. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 847–852. IEEE, 2021.

[81] Shipeng Yu, Faisal Farooq, Alexander Van Esbroeck, Glenn Fung, Vikram Anand, and Balaji Krishnapuram. Predicting readmission risk with institution-specific prediction models. *Artificial Intelligence in Medicine*, 65(2):89–96, 2015.

[82] Shuai Yu, Xu Chen, Zhi Zhou, Xiaowen Gong, and Di Wu. When deep reinforcement learning meets federated learning: Intelligent multi-timescale resource management for multi-access edge computing in 5G ultra dense network. *IEEE Internet of Things Journal*, 2020.

[83] Yehong Zhang, Zhongxiang Dai, and Bryan Kian Hsiang Low. Bayesian optimization with binary auxiliary information. In *Proc. UAI*, pages 1222–1232, 2020.

[84] Yehong Zhang, Trong Nghia Hoang, Bryan Kian Hsiang Low, and Mohan Kankanhalli. Information-based multi-fidelity Bayesian optimization. In *Proc. NeurIPS Workshop on Bayesian Optimization*, 2017.

[85] Zhaowei Zhu, Jingxuan Zhu, Ji Liu, and Yang Liu. Federated bandit: A gossiping approach. In *Abstract Proceedings of the 2021 ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems*, pages 3–4, 2021.

[86] Hankz Hankui Zhuo, Wenfeng Feng, Qian Xu, Qiang Yang, and Yufeng Lin. Federated reinforcement learning. arXiv:1901.08277, 2019.