UNIVERSITY OF CALIFORNIA SAN DIEGO

Uncertainty Reduction in Supervised Learning and Online Decision Making

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Xinghan Wang

Committee in charge:

Professor Tara Javidi, Chair
Professor Sanjoy Dasgupta
Professor Sicun Gao
Professor Farinaz Koushanfar

2024

The Dissertation of Xinghan Wang is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

DEDICATION

To my wife and life partner, who is also my best friend and is always by my side.

EPIGRAPH

Life is nothing until it is lived;
but it is yours to make sense of,
and the value of it is nothing else but the sense that you choose.

Jean-Paul Sartre – Existentialism Is a Humanism

TABLE OF CONTENTS

Dissertation Approval Page iii

Dedication iv

Epigraph v

Table of Contents vi

List of Figures viii

List of Tables x

Acknowledgements xi

Vita xiii

Abstract of the Dissertation xiv

Chapter 1 Introduction 1
1.1 Federated Learning 1
1.2 Reinforcement Learning 4
1.2.1 Multi-armed and Contextual Bandits 5
1.2.2 Markov Decision Processes (MDP) 8
1.2.3 Linear Quadratic Regulators (LQR) 9
1.3 Multi-Agent Reinforcement Learning Problem 10
1.4 Research Overview 11

Chapter 2 Peer-to-Peer Variational Federated Learning over Graphs 14
2.1 Introduction 15
2.2 The Model: Decentralized Learning 19
2.3 Peer-to-peer Variational Learning over an Arbitrary Graph 23
2.4 Main Results 25
2.5 Experiments 29
2.5.1 Decentralized Bayesian Linear Regression 29
2.5.2 Decentralized Image Classification 30
2.6 Conclusion and Future Work 36
2.7 Acknowledgements 37

Chapter 3 Contextual Shortest Path: a Centralized RL problem 38
3.1 Introduction 39
3.2 Problem formulation 41
3.3 Algorithms for Contextual Shortest Path 42
3.3.1 Known context distributions $\mathbf{P}_i^a$ 42
3.3.2 Unknown context distributions $\mathbf{P}_i^a$ 43

3.4 Experiments 46
3.5 Conclusion 53
3.6 Acknowledgements 54

Chapter 4 Federated Certainty Equivalence Control: a Decentralized RL problem . . . 55
4.1 Introduction 55
4.1.1 Notation 58
4.2 Preliminaries 58
4.2.1 Centralized LQ control with known system model 58
4.2.2 Centralized Adaptive LQ control with unknown system model 59
4.3 Problem Statement: Decentralized LQ control with unknown system model 60
4.4 Adaptive Algorithm for DecLQ 63
4.4.1 FedCE Algorithm 64
4.5 Convergence and Regret Analysis 66
4.5.1 Satisfying Rich and Stable Exploration 71
4.6 Numerical Analysis 72
4.6.1 Zero-target control 72
4.6.2 Beyond zero-target Decentralized LQ Problems 76
4.7 Conclusions and Future Work 84
4.8 Acknowledgements 84

Chapter 5 Conclusion and Future Directions 86

Appendix A Appendix to Chapter 2 88
A.1 Bernstein-Von Mises Theorem under model misspecification 88
A.2 Consensus Step on Gaussian distributions 89
A.3 Proof of Theorem 1 89
A.4 Proof of Corollary 1 92

Appendix B Appendix to Chapter 3 93

Appendix C Appendix to Chapter 4 95

Bibliography 96

LIST OF FIGURES

Figure 1.1. Typical federated learning setup in a text-completion mobile app scenario [127] 2

Figure 1.2. Illustration of federated learning (left) and decentralized learning (right) . . 4

Figure 1.3. Illustration of Agent-Environment interaction in Reinforcement Learning [147] 5

Figure 2.1. Mean Squared Error (MSE) of the predictions over a test dataset under two cases: (i) all agents, despite the severe deficiency of their observations, learn without cooperation using local training data only, and (ii) agents learn using the proposed decentralized learning rule. 30

Figure 2.2. Figure shows average accuracies over all nodes of proposed decentralized learning algorithm and FedAvg on FEMNIST dataset. 32

Figure 2.3. Figure shows the variation in the average accuracy over a star network topology as the eigenvector centrality of the central agent is varied. 34

Figure 2.4. Figure shows the accuracies of selected agents in a time-varying network. 35

Figure 3.1. Network topologies. s denotes source and d denotes destination. For (b) and (c), the red route is the expected optimal route. For (d) the red edges are those with smaller expected weights. 49

Figure 3.2. $(\mathbf{a})(\mathbf{b})(\mathbf{c})$ Regret under different network topologies for CSP. $(\mathbf{d})$ Regret for SSP. 50

Figure 4.1. Performance of FedCE and other alternative algorithms. (a) Mean Regret, (b) Model Estimation Error. 74

Figure 4.2. Ablation studies of FedCE under different $(\mathbf{a})$ scaling factor k, and $(\mathbf{b})$ initial excitation a^1. 76

Figure 4.3. Convergence of x_1 and x_2 to each other for various R. 77

Figure 4.4. Performance of FedCE for an unstable system. 78

Figure 4.5. Trajectory of two agents in the case of nonzero reference. 80

Figure 4.6. Cumulative regret in the case of nonzero reference. 81

Figure 4.7. Trajectory of agent 1 in the case of tracking a trajectory generated from a known linear system. 83

Figure 4.8. Trajectory of agent 2 in the case of tracking a trajectory generated from a known linear system. 83

LIST OF TABLES

Table 2.1. Settings for Star Topology Network Experiment 34

Table 2.2. Settings for Time-varying Network Experiment 36

ACKNOWLEDGEMENTS

I would like to thank my Ph.D. advisor Professor Tara Javidi for her support as the chair of my committee. I would not be able to navigate this journey without her valuable insights and encouragement. Many thanks to my committee members, Professor Sanjoy Dasgupta, Professor Sicun Gao and Professor Farinaz Koushanfar, for their valuable suggestions and guidance.

I would also like to thank my wife, Xinyue, who is also my best friend and life partner. To say thank you is such an understatement. Knowing there is always someone to whom I can talk is such a relief, especially during dark times such as COVID-19.

Thanks to my parents who supported me since I was an undergrad in the U.S. Thank you for your unremitting support of my education and the many opportunities you have provided me.

Thanks to my co-authors, Gregory Fields, Nasimeh Heydaribeni and Anusha Lalitha, for all the useful discussions and collaborations. Many thanks to my labmates, who accompanied me throughout this painstaking yet rewarding journey, especially Chi-shiang Gau and Xinran Zheng. Thank you all for your mentorship, friendship, and collaboration over the years, for which I am deeply grateful.

Last but not least, I have also been blessed with a fantastic badminton community here in San Diego. I will cherish all the good games we played, tournaments we teamed up to, and good spirits and vibes in the gym. I will surely miss this badminton community dearly.

Chapter 2, in full, is a reprint of the material published in the IEEE Journal on Selected Areas in Information Theory (JSAIT), vol. 3, no. 2, pp. 172-182, June 2022: "Peer-to-Peer Variational Federated Learning over Arbitrary Graphs" (Xinghan Wang, Anusha Lalitha, Tara Javidi, and Farinaz Koushanfar). The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of the material published in the 55th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, pp. 471-476, 2021: "Contextual Shortest Path with Unknown Context Distributions" (Xinghan Wang, Gregory Fields, and Tara Javidi). The dissertation author was the primary investigator and author of this paper.

Chapter 4, in part, is a reprint of the material published in the 59th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 2023, pp. 1-8: "Federated Certainty Equivalence Control for Linear Gaussian Systems with Unknown Decoupled Dynamics and Quadratic Common Cost" (Xinghan Wang, Nasimeh Heydaribeni, Farinaz Koushanfar and Tara Javidi). The dissertation author was the primary investigator and author of this paper.

VITA

2014–2018 Bachelor of Science in Electrical Engineering, Washington University in St. Louis

2017–2018 Master of Science in Electrical Engineering, Washington University in St. Louis

2018–2024 Doctor of Philosophy in Computer Science, University of California San Diego

PUBLICATIONS

[1] "Decentralized Bayesian Learning over Graphs". A. Lalitha, X. Wang, O. C. Kilinc, Y. Lu, T. Javidi, and F. Koushanfar. *ArXiv, abs/1905.10466, 2019*.
[2] "Contextual Shortest Path with Unknown Context Distributions". X. Wang, G. Fields, and T. Javidi. 55th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, pp. 471-476, 2021.
[3] "Peer-to-Peer Variational Federated Learning over Arbitrary Graphs". X. Wang, A. Lalitha, T. Javidi, and F. Koushanfar. IEEE Journal on Selected Areas in Information Theory (JSAIT), vol. 3, no. 2, pp. 172-182, June 2022.
[4] "Federated Certainty Equivalence Control for Linear Gaussian Systems with Unknown Decoupled Dynamics and Quadratic Common Cost". X. Wang, N. Heydaribeni, F. Koushanfar, and T. Javidi. 2023 59th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 2023, pp. 1-8.

ABSTRACT OF THE DISSERTATION

Uncertainty Reduction in Supervised Learning and Online Decision Making

by

Xinghan Wang

Doctor of Philosophy in Computer Science

University of California San Diego, 2024

Professor Tara Javidi, Chair

In machine learning, there are two primary types of uncertainty: aleatoric uncertainty, reflecting inherent noise in observations, and epistemic uncertainty, pertaining to model uncertainty which can be reduced with more data. In both supervised and reinforcement learning tasks, understanding and managing these uncertainties is vital for improving model performance and reliability. In this dissertation, we study three cases of uncertainty reduction, from supervised learning to reinforcement learning, and cover the centralized and the decentralized cases.

We first investigate decentralized learning of supervised tasks using variational Bayesian deep networks. In this setup, agents on a peer-to-peer network collaboratively learn a global model while maintaining potentially non-IID local data. The chapter demonstrates that each agent

eventually learns the true model parameters, achieving accurate parameter estimates based on the structure of the communication network and the agents' relative learning capacity. Theoretical analyses reveal that modeling epistemic uncertainty enhances generalization capabilities in decentralized settings, where knowledge is shared without centralized data aggregation.

We then turn to RL problems involving aleatoric uncertainty, particularly due to unknown stochastic transitions in environments. We first introduce the Contextual Shortest Path (CSP) problem, inspired by dynamic path planning applications such as UAVs navigating stochastic, context-dependent paths. In this episodic MDP, an agent must learn to navigate a graph with random, context-dependent edges. To manage aleatoric uncertainty, two baseline algorithms—*Thompson Sampling* and ε*-greedy*—are adapted for this setting, followed by a proposed algorithm, RL-CSP, which optimizes exploration across time steps, ensuring under-explored states are visited efficiently. A theoretical bound on regret for RL-CSP is derived, and simulations are presented to validate the algorithm's performance across network topologies.

Next, we address multi-agent RL in decentralized linear quadratic (LQ) control, where agents operate in partially observable linear Gaussian systems with unknown transition dynamics. Here, aleatoric uncertainty arises from unknown system properties and partial state observations. We present an algorithm based on Certainty Equivalence that alternates between the exploration and exploitation phases. Regret bounds are established, and extensive simulations under various scenarios illustrate the effectiveness of this approach.

Chapter 1

Introduction

In this chapter, we will first give a high-level introduction to Federated Learning and Reinforcement Learning problems, which serve as background to the main thesis. We will then discuss the Multi-agent Reinforcement Learning problem as well as the structure of the remaining chapters.

1.1 Federated Learning

Federated learning is a decentralized approach to machine learning that enables models to be trained across multiple devices or servers without requiring them to transfer their data to a central repository. In traditional machine learning, data from multiple sources is typically collected and centralized on one server, where the model is trained. Federated learning, however, flips this concept by allowing each device—whether a smartphone, IoT device, or computer—to train a local version of the model using its own unique data. Once each device completes training, it sends the model's learned updates (instead of the actual data) back to a central server. The central server then aggregates these updates from many devices to improve a shared, global model, effectively learning from distributed data without ever centralizing it. Figure 1.1 illustrates a typical federated learning workflow in a mobile application scenario.

This method offers several significant advantages. First, it enhances user privacy by keeping data on users' devices and transmitting only the model updates, not raw data. This is

Figure 1.1. Typical federated learning setup in a text-completion mobile app scenario [127]

particularly beneficial for applications in sensitive areas, such as healthcare [161, 126], finance [112], or personalized mobile edge services [154, 106], where data privacy and security are critical. Since federated learning minimizes the need to collect data in a central location, it reduces risks associated with data breaches or misuse.

Additionally, federated learning can lower data transmission costs. In contexts where internet bandwidth or power is limited, such as mobile devices in remote areas, reducing the need to upload large datasets makes the system more efficient and accessible. This decentralized approach also supports faster adaptation to local data patterns, which means models can become more relevant and accurate for specific user groups or geographic regions.

A prototypical federated learning setup consists of a central server and K clients. Each client k has access to a local dataset $\mathscr{D}_k$, where $|\mathscr{D}_k| = n_k$. The learning objective of federated learning is to find w^* that is shared by all clients and is the minimizer of the following objective:

$$w^* = \underset{w}{\operatorname{argmin}} f(w) := \sum_{k=1}^{K} p_k F_k(w)$$

where $F_k(\cdot)$ is the local objective function for client k. Usually, this is defined as the empirical risk calculated over the private training set $\mathscr{D}_k$:

$$F_k(w) = \frac{1}{n_k} \sum_{(x_i, y_i) \in \mathscr{D}_k} l(x_i, y_i; w)$$

where $l(\cdot)$ is an instance-level loss (e.g., cross-entropy loss or squared error in the case of classification or regression tasks, respectively). Furthermore, each $p_k \geq 0$ specifies the relative contribution of each client. Possible settings for p_k are $p_k = \frac{n_k}{n}$, where $n = \sum_{k=1}^{K} n_k$, or $p_k = \frac{1}{K}$.

The generic federated round at each time t is decomposed into the following steps and iteratively repeated until convergence [115], i.e. for $t = 1, 2, ...$:

1. **Client selection**: K clients selected based on availability, etc
2. **Broadcast**: selected clients download global weight w_t
3. **Local computation**: clients update local model (e.g. SGD), $w_{t+1}^k = w_t - \eta \nabla F_k(w_t)$
4. **Server aggregation**: collects clients' model weights w_{t+1}^k or gradient $\nabla F_k(w_t)$.
5. **Model update**: $w_{t+1} = \sum_{k=1}^{K} p_k w_{t+1}^k$, or $w_{t+1} = w_t - \eta \sum_{k=1}^{K} p_k \nabla F_k(w_t)$

Federated learning comes with unique challenges, such as handling diverse data sources or often called Non-IID data distributions (since each device may have very different data) [20, 168, 102, 62], ensuring efficient communication of updates [39, 140, 135], and managing secure aggregation methods to protect privacy [101, 157, 120].

Remark 1 (Decentralized Learning)**.** *Decentralized learning is an extension of federated learning, in that data is still distributed on clients' devices. However, federated learning uses star topology, which puts a communication bottleneck on a central server. Decentralized Learning allows peer-to-peer communication among clients, therefore does away with the need for a centralized server (Figure 1.2). Usually, a doubly stochastic communication matrix W is defined, and clients update their local models based on weighted average of neighbors' model/gradient, e.g. $w^k = \sum_{j \in \mathcal{N}_k} W_{i,j} w^j$. Decentralized learning faces similar challenges to federated learning, including Non-IID data distribution [55, 75, 149], asynchronous local updates [130, 133], privacy [166, 55, 75] and adversarial attacks/data poisoning [22, 66].*

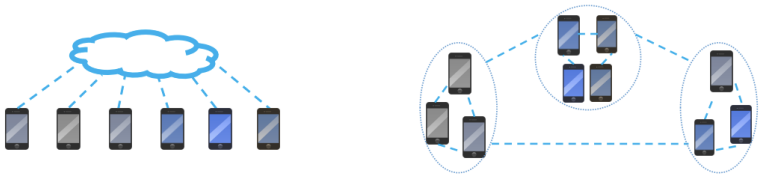


Figure 1.2. Illustration of federated learning (left) and decentralized learning (right)

1.2 Reinforcement Learning

We now turn our attention to Reinforcement Learning (RL), which is another component of the thesis. The intuition of RL is to train decision-making entities, called "agents", to learn by interacting with the environment instead of learning from labeled examples as in traditional supervised learning. Here, an agent learns by interacting with an environment, making decisions, and receiving feedback in the form of rewards or penalties. Over time, this feedback helps the agent understand which actions lead to success and which do not.

On a high level, the fundamental components of reinforcement learning include an agent, an environment, actions, rewards, and states. The agent is the decision-maker, while the environment represents everything it interacts with. Actions are choices the agent can make, and the state represents the current situation of the environment. After each action, the agent receives a reward—a numerical score indicating how beneficial or harmful the action was toward achieving its goal. By repeatedly interacting with the environment, the agent's goal is to maximize its cumulative reward over time, often by developing a strategy or policy to make the best possible decisions in any given state (See illustration in Figure 1.3).

Reinforcement learning is unique in its ability to solve complex, sequential decision-making problems, making it particularly useful for applications where outcomes are based on a series of actions rather than a single decision. For instance, RL is widely used in robotics, where agents learn to perform tasks like grasping objects or walking [83, 132]. It's also been instrumental in game playing, such as Atari [74] and the groundbreaking success of AlphaGo

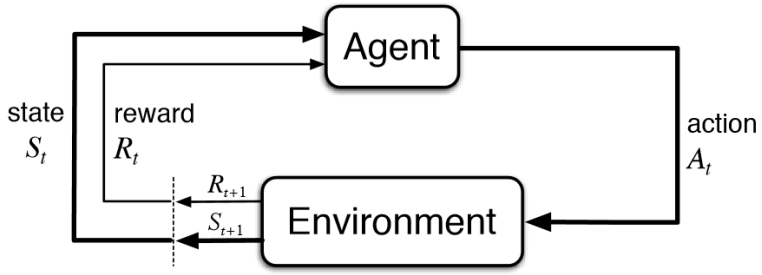


Figure 1.3. Illustration of Agent-Environment interaction in Reinforcement Learning [147]

[143], which mastered the game of Go by playing millions of games against itself. Other fields, like finance [57], healthcare [163], and autonomous driving [81], also benefit from reinforcement learning as it can handle dynamic and uncertain environments.

1.2.1 Multi-armed and Contextual Bandits

Perhaps the most classic and simplest (in terms of description, not to solve) RL problem is the Multi-armed Bandit (MAB) problem. Initially introduced by William R. Thompson in an article published in 1933 in Biometrika [152], the MAB problem was revisited in the 1950s [31] to study learning behavior in humans, where a 'two-armed bandit' machine was commissioned where humans could choose to pull either the left or the right arm of the machine, each giving a random pay-off with the distribution of payoffs for each arm unknown to the human player. The goal is to play the slot machine in a way that maximizes total winnings over time.

What would be the strategy to play this machine? One could play each arm of the machine a number of times, and analyze the observed payoffs before moving forward with more pulls. But how many times should one play before deciding an arm is better than the other? Should one explore an option that looks inferior or exploit by going with the option that looks best currently? This dilemma captures the one of the fundamental challenges in RL: balancing exploration v.s. exploitation [147].

Formally, in a k-armed bandit problem, each of the k actions has an unknown mean reward of μ_k if that arm is pulled. Suppose we play the machine T times. For time step $t \in [T]$,

an action $A_t \in \mathscr{A}$ is taken, where $\mathscr{A}$ is the *action space* which refers to any of the k arms, and a *reward* $R_t \in \mathbb{R}$ is revealed. Denote $q^*(a)$ as the *value* of action a, which is the expected reward given that a is selected, then: $q^*(a) := \mathbb{E}[R_t|A_t = a] = \mu_a$. If $q^*(a)$ is known, we could always select $A_t = \text{argmax}_{a\in\mathscr{A}} q^*(a)$. But since it is unknown, we would need to estimate $q^*(a)$ empirically, i.e. from observations of rewards of each arm. We denote that estimate as $Q_t(a)$. One straightforward way of computing $Q_t(a)$ is using the empirical mean of each arm, i.e.

$$Q_t(a) = \frac{\sum_{i=1}^{t} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t} \mathbb{1}_{A_i=a}}$$

At any time step, we call the one action whose $Q_t(a)$ is greatest the *greedy action*. When you select the *greedy action*, we say that you are *exploiting* your current knowledge of the values of the actions. If instead, you select one of the non-greedy actions, then we say you are *exploring* because this enables you to improve your estimate of the non-greedy action's value. *Exploitation* is the right thing to do to maximize the expected reward on the one step, but *exploration* may produce a greater total reward in the long run. Now we can define *regret* as:

$$\mathscr{R}(t) = \sum_{i=1}^{t} \left[\max_{a\in\mathscr{A}} q^*(a) - R_t \right]$$

Then our goal of maximizing cumulative reward is equivalent to minimizing *regret*.

One way to balance exploration and exploitation in MAB is called "*Explore-then-Commit* (ETC)" [136], which explores all arms uniformly at the beginning before committing to one. Another way is called "ε-*greedy*" [147], where we select the greedy action most of the time but take a little exploration with probability ε. A more involved and influential algorithm, "*Upper-Confidence-Bound* (UCB)" [88, 89, 13], employs the intuition that it would be better to select among the non-greedy actions according to their potential for actually being optimal, formally: $A_t = \text{argmax}_{a\in\mathscr{A}} \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$, where $N_t(a)$ is the number of times a is selected up to t. Another line of randomized exploration strategies is *probability matching* or *posterior sampling*,

now more commonly known as "*Thompson Sampling*" [152, 139, 36], reflects the idea that the number of pulls for a given arm should match its actual probability of being the optimal arm.

Stochastic Contextual Bandits, or usually just referred to as Contextual Bandits [15, 151], is an extension to MAB. In contextual bandits, before choosing an action A_t, one is presented with a random *context* $C_t \in \mathscr{C}$. Given this context, one selects action A_t and receives reward R_t in the form:

$$R_t = r(C_t, A_t) + \eta_t$$

where $r(\cdot,\cdot) : \mathscr{C} \times \mathscr{A} \to \mathbb{R}$ is the *reward function* and η_t is some noise. If r is given, then best action in each round is: $A_t = \text{argmax}_{a \in \mathscr{A}} r(C_t, a)$. In this setting, you are essentially facing multiple MABs: on observing C_t, you decide which MAB you are likely facing and select the best arm corresponding to that MAB. In other words, the action you choose is conditioned on context, $A_t = \pi(a|C_t)$. In RL, $\pi : \mathscr{C} \to \mathscr{A}$ is called *policy*. The regret for contextual bandits is therefore defined as:

$$\mathscr{R}(t) = \mathbb{E}\left[\sum_{i=1}^{t} \max_{a \in \mathscr{A}} r(C_t, a) - \sum_{i=1}^{t} R_t\right]$$

where the expectation is taken over the randomness of context and noise. One way to eventually learn an optimal policy is to estimate $r(c,a)$ for each (c,a) pair, but that space can get prohibitively large. A simplifying assumption can be made that there exists a *feature map* $\psi : \mathscr{C} \times \mathscr{A} \to \mathbb{R}^d$, and for an unknown parameter $\theta^* \in \mathbb{R}^d$, it holds that:

$$r(c,a) = \langle \theta^*, \psi(c,a) \rangle$$

This gives rise to Stochastic Linear Bandits [3], where in round t, one chooses an action

$A_t \in \mathscr{A}$, and receives reward:

$$R_t = \langle \theta^*, A_t \rangle + \eta_t$$

Note that if $\mathscr{A} = \{e_1, ..., e_d\}$, the linear bandit reduces to a MAB. On the other hand, if $\mathscr{A} \subseteq \{0,1\}^d$, then we have a Combinatorial Bandit [35]. Similar to MAB, where one estimates $q^*(a)$ while maximizing cumulative rewards, linear bandit usually requires estimating unknown vector θ^*. Due to this parallelism, intuitions gained from MAB can usually be transferred to linear bandits, including UCB-type algorithms [47, 14, 72], Thompson Sampling style algorithms [6] and *Epoch-greedy* [97].

1.2.2 Markov Decision Processes (MDP)

In contextual bandits, for round t, the agent sees context C_t, takes action A_t and receives reward R_t; then for round $t+1$, a new context C_{t+1} is presented, independent from C_t, and the process repeats. What if C_{t+1} is dependent on C_t and A_t? This motivates us to consider a more complex kind of RL problem: Markov Decision Processes (MDPs).

A Markov Decision Process (MDP) is usually defined by the 4-*tuple*: $(\mathscr{S}, \mathscr{A}, r, p)$, where $\mathscr{S}$ is called the *state space*, $\mathscr{A}$ the *action space*, $r : \mathscr{S} \times \mathscr{A} \to \mathbb{R}$ the *reward function*, and $p : \mathscr{S} \times \mathscr{A} \times \mathscr{S} \to \mathbb{R}$ the *transition probabilities*. At time step t, the agent observes current state $S_t \in \mathscr{S}$, takes action $A_t \in \mathscr{A}$, and receives reward $R_t = r(A_t, S_t)$. It then transitions to the next state S_{t+1}, according to $p(S_{t+1}|S_t, A_t)$. The goal is to maximize cumulative rewards, as in MAB and contextual bandits. For *episodic tasks*, such as plays of a game or trips through a maze, each episode ends in a special state called the *terminal state*. For *continuing tasks*, such as process-control, there is no finite time horizon, in order for the objective to be meaningful a *discount factor* γ is introduced so that the sum of the discounted rewards it receives over the future is maximized.

Two important notions in MDPs are *policies* and *value functions*. Similar to the policy

defined in contextual bandits, the policy for an MDP is a mapping from states to probabilities of selecting each possible action. If the agent is following policy π at t, then $\pi(a|s)$ is probability that $A_t = a$ if $S_t = s$. Given a policy π, the *value function* $v_\pi(s)$ of a state s, is the expected reward if the agent starts in s and follows π afterward. It is also useful to define the *action-value function*, $q_\pi(s,a)$, which is the expected return of taking action a in state s and following π afterward. The optimal policy is therefore: $\pi^* = \text{argmax}_\pi v_\pi(s)$ for state s or $\pi^* = \text{argmax}_\pi q_\pi(s,a)$ for state-action pair (s,a).

When transition probabilities and reward functions are known, and state/action spaces are finite, optimal policies to achieve maximum reward can be derived through *Dynamic Programming* type algorithms utilizing Bellman recursions, namely *policy iteration* and *value iteration* [147], where value functions are approximated iteratively. These algorithms are called *model-based* since information about the underlying MDP (i.e. the *model*) is required. Another line of work named *Q-learning* employs the idea from *temporal-difference learning* (TD) and directly approximates the action-value function q, and does away with the need for model parameters and thus called *model-free*. Examples include *Q-learning* [155], *SARSA* [138, 146]; when dealing with infinite state/action spaces, deep networks are involved for function approximation, resulting in works like *Deep Q-Networks* (DQN) [116], *Deep Deterministic Policy Gradient* (DDPG) [105] and *TD3* [49].

It is worth mentioning that for finite state/action spaces (sometimes called *tabular MDPs*), when transition probabilities and reward functions are unknown, there are *model-based* algorithms that learn the model online while minimizing regret. Examples include *Explicit Explore or Exploit* (E^3) [76], UCB-type algorithms *R-Max* [29], *UCRL* [17] and *UCRL2* [16], and *Thompson Sampling* style algorithms like [129].

1.2.3 Linear Quadratic Regulators (LQR)

The MDP problem is deeply rooted in the field of *optimal control* [86, 87]. A simple yet well-studied problem in optimal control is called Linear Quadratic Regulator (LQR). In

discrete-time, the system dynamics for LQR are often represented as:

$$x(t+1) = Ax(t) + Bu(t) + w(t), w(t) \textit{ i.i.d. noise} \sim \mathcal{N}(0, \sigma_w I)$$

with *step cost* defined as (Q, R are positive semi-definite):

$$c(t) = x(t)^T Q x(t) + u(t)^T R u(t)$$

The goal is to find $\pi : \mathscr{X} \to \mathscr{U}$, that minimizes the *infinite horizon averaged cost*: $J^\pi(\theta) = \text{limsup}_{T\to\infty} \frac{1}{T}\mathbb{E}^\pi[\sum_{t=1}^T c(t)]$. Notice the parallel between MDP and LQR: if we write $P(x_{t+1}|x_t, u_t) = \mathcal{N}(Ax_t + Bu_t, \sigma_w I)$, $r(x_t, u_t) = x_t^T Q x_t + u_t^T R u_t$, then LQR is essentially an MDP with infinite state/action space and infinite horizon. It is known that the cost-minimizing control takes the form: $u(t) = -K(\theta)x(t)$, where $K(\theta)$ is derived from Algebraic Riccati Equation (ARE) [42].

When system dynamics $\theta^* = (A, B)$ are unknown, RL algorithms are involved in estimating θ^* while minimizing costs. For *model-free* approaches, [63] utilizes policy gradient and [113] considers zeroth-order methods. For *model-based* approaches, there exist three lines of work: *Certainty equivalence (CE)*, *Optimism in the Face of Uncertainty (OFU)* i.e. UCB-type, and *Thompson Sampling (TS)*. *Certainty equivalence (CE)* probes system with noise with diminishing variance and estimates θ^* [19, 38, 46]. *Optimism in the Face of Uncertainty (OFU)* maintains a confidence ellipsoid around θ^* and selects the most "optimistic" model [1, 40, 45, 65, 90]. Lastly, *Thompson Sampling (TS)* puts prior over θ^* and samples model from the posterior [4, 11, 52].

1.3 Multi-Agent Reinforcement Learning Problem

Multi-Agent Reinforcement Learning (MARL) is an extension of reinforcement learning where multiple agents interact with each other within a shared environment. In traditional (centralized) reinforcement learning, a single agent learns to make decisions in isolation, aiming

to maximize its cumulative reward. In MARL, each agent's actions and decisions not only affect its own outcomes but also potentially influence the experiences and rewards of other agents in the environment. This introduces a layer of complexity, as agents must learn to adapt to each other while striving to achieve their individual or shared goals.

In MARL, agents can be cooperative [109, 98], where agents work together toward a common goal, like coordinating robots to move objects in a warehouse. They can be competitive [108], where agents compete with each other, trying to maximize their individual rewards, as in many games like chess or soccer. Or they can be a mixture of both [144], for instance in a driving scenario, cars (agents) may act in their self-interest but must cooperate to avoid accidents.

Virtually all the aforementioned single agent RL problems have their multi-agent counterpart: from multi-agent MAB (MAMAB) [96, 142], to multi-agent contextual/linear bandits [150], to multi-agent MDP [70, 171], and to multi-agent LQR [11, 51]. MARL faces the same exploration/exploitation dilemma as single agent RL, plus its own challenges, including scalability, effective communication strategies, and nonstationarity. Like federated learning, these challenges are common for a decentralized machine learning system. For a complete overview of MARL, please see [32] and more recent [165].

1.4 Research Overview

In machine learning, there are two major types of uncertainty one can model: *Aleatoric uncertainty* and *epistemic uncertainty*. Aleatoric uncertainty, also called statistical uncertainty, captures noise inherent in the observations. On the other hand, epistemic uncertainty accounts for uncertainty in the model – uncertainty which can be explained away given enough data [78].

For supervised learning tasks, recent advances in Bayesian Neural Networks (BNN) make it possible to capture epistemic uncertainty [25]. In BNN, epistemic uncertainty is modeled by placing a prior distribution over a deep learning model's weights, and then trying to capture how much these weights vary given some data. For reinforcement learning problems, aleatoric

uncertainty originates from the stochastic nature of the environment and interactions with the environment. There are 3 main potential sources of aleatoric uncertainty in reinforcement learning (effectively one for each component of the MDP): stochastic rewards, stochastic observations, and stochastic actions. The stochastic observations can stem from incomplete observations or stochastic transition dynamics. If the transition function p in the MDP is nondeterministic, then the transition from one state to the next is a source of aleatoric uncertainty [111].

In this dissertation, we study three cases of uncertainty reduction, from supervised learning to reinforcement learning, and cover the centralized and decentralized (federated) cases.

In Chapter 2, we consider decentralized learning of supervised tasks with variational Bayesian deep networks, over a peer-to-peer communication graph. We show that for the realizable case, each agent will eventually learn the true parameters associated with the global model, with potentially non-IID local data distributions. We further provide an analytical characterization of the rate of convergence of the posterior probability, essentially epistemic uncertainty for model weights, at each agent in the network as a function of network structure and local learning capacity as measured by the relative entropy. Empirically, allowance of epistemic uncertainty enables better generalization capabilities in a decentralized learning setting.

In Chapter 3 and 4, we turn our attention to reinforcement learning problems, focusing on uncertainties originating from unknown stochastic transition dynamics. In Chapter 3, we study a new class of episodic MDP problems, which we call the Contextual Shortest Path (CSP) problems. The problem is motivated by dynamic path planning and obstacle avoidance for UAV and drone applications. An agent navigates an undirected graph with stochastic edges, where random edge realizations are augmented with richer contexts that can change over time. Aleatoric uncertainty in this case is from unknown stochastic context distributions given each state-action pair. When the context distributions are unknown and need to be learned online, we first adapt two algorithms as our baselines, one based on Thompson Sampling and the other based on ε-greedy exploration. We then propose a novel reinforcement learning algorithm, RL-CSP, which intelligently distributes exploration episodes over the time horizon and ensures the agent

visits under-explored states. We bound the regret for RL-CSP, and augment the theoretical results with simulations over various network topologies.

In Chapter 4, we study a class of Multi-Agent Reinforcement Learning (MARL) problem called the decentralized linear quadratic (LQ) control problem. In this case, aleatoric uncertainty stems from unknown transition dynamics and partial state observations in a linear Gaussian system. In our proposed algorithm, we partition time steps into exploitation and exploration intervals. During the exploration phase, we employ Least Square Estimation (LSE) techniques to obtain local partial system model estimates. These estimates are then shared between agents at the end of each exploration interval. Subsequently, both agents compute Certainty Equivalence controllers, which they apply during the exploitation interval. We analyze the algorithm in terms of its regret bound, and provide extensive numerical analysis under three scenarios that support our theoretical results.

Chapter 2

Peer-to-Peer Variational Federated Learning over Graphs

In this chapter, we consider decentralized learning of supervised tasks. We propose a federated supervised learning framework over a general peer-to-peer network with agents that act in a variational Bayesian fashion. The proposed framework consists of local agents where each of which keeps a local *"posterior probability distribution"* over the parameters of a global model; the updating of the posterior over time happens in a local fashion according to two subroutines of: 1) variational model training given (a batch of) local labeled data, and 2) asynchronous communication and model aggregation with the 1-hop neighbors. Inspired by the popular federated learning (model averaging), the framework allows the training data to remain distributed on mobile devices while utilizing a peer-to-peer model aggregation in a social network.

The proposed framework is shown to allow for a systematic treatment of model aggregation over any arbitrary connected graph with consistent (in general, non-iid) local labeled data. Specifically, under mild technical conditions, the proposed algorithm allows agents with local data to learn a shared model explaining the global training data in a decentralized fashion over an arbitrary peering/connectivity graph. Furthermore, the rate of convergence is characterized and shown to be a function of each individual agent's data quality weighted by its eigenvector centrality. Empirically, the proposed methodology is shown to work well with efficient variation

Bayesian inference techniques to train Bayesian neural networks in a decentralized manner even when the local data batches are not identically distributed.

2.1 Introduction

Personal edge devices can often use their locally observed data to learn machine learning models that improve the user experience. However, the use of local data for learning globally rich machine learning models has to address two important challenges. First of all, this type of localized data, in isolation from the data collected by other devices, might be statistically insufficient to learn a global model. Secondly, there might be severe restrictions on sharing raw forms of personal/local data due to privacy and communication cost concerns. In light of these challenges and restrictions, an alternative approach has emerged which leaves the training data distributed on the edge devices while enabling the decentralized learning of a shared model. This alternative, known as *Federated Learning*, is based on edge devices' periodic communication with a central (cloud-based) server responsible for iterative model aggregation. While addressing the privacy constraints on raw data sharing, and significantly reducing the communication overload as compared to synchronized stochastic gradient descent (SGD), this approach falls short in fully decentralizing the training procedure. Many practical peer-to-peer networks are dynamic and a regular access to a fixed central server, which coordinates the learning across devices, is not always possible. Existing methods based on federated learning cannot handle such general networks where central server is absent and/or when the data has severe heterogeneity across the network.

To summarize, some of the major challenges encountered in a fully decentralized learning paradigm are: (i) *Statistical Insufficiency and non-IID Data Distributions:* The local and individually observed data distributions are likely to be less rich than the global training set. For example, a subset of features associated with the global model may be missing locally. (ii) *Restriction on Data Exchange:* Due to privacy concerns, agents do not share their raw training

data with the neighbors. Furthermore, model parameter sharing has been shown to reduce the communication requirements significantly. (iii) *Lack of Synchronization:* There may not be a single agent with whom every agent communicates, which can synchronize the learning periodically. (iv) *Localized Information Exchange:* Agents are likely to limit their interactions and information exchange to a small group of their peers which can be viewed as the 1-hop neighbors on the social network graph. Furthermore, information obtained from different peers might be viewed differently, requiring a heterogeneous model aggregation strategy.

Contributions: We consider a fully decentralized learning paradigm where agents iteratively update their models using local data and aggregate information from their neighbors to their local models. In particular, we consider a learning rule where agents take a variational (Bayes) learning approach via the introduction of a posterior distribution over a parameter space characterizing the unknown global model.

Our contributions are as follows:

1. On the algorithmic side, our decentralized learning rule is inspired by works on social learning, distributed hypothesis testing literature [94, 141, 124, 92]. Our social learning rule builds on our recent work on federated log posterior averaging [95, 93] and generalizes posterior averaging proposed in [7].

2. We provide the first theoretical guarantees for the realizable case where the hypothesis (parametric model) class contains the true labeling function.

    - We prove that, under mild technical assumptions, each agent will eventually learn the true parameters associated with global model, with potentially non-IID local data distributions.

    - We provide analytical characterization of the rate of convergence of the posterior probability at each agent in the network as a function of network structure and local learning capacity as measured by the relative entropy.

3. Unlike prior work, we allow a general network structure as well as non-IID data distributions. As a consequence, our work provides first known theoretical guarantees on convergence for a variational federated learning on an arbitrary connected graph.

4. In addition to our theoretical results, we investigate the performance of our proposed variational learning empirically.

    - We show that the practical advantage of our approach for non-IID data over the classical federated averaging especially as the complexity of the task increases.
    - We illustrate the impact of social network structure on the model convergence.
    - We also demonstrate the ability of our approach to scale up in a time-varying asynchronous network.
    - We demonstrate the ability of our approach to deal with all 4 cases of non-IID local data.

In this regard, our work bridges the theoretical gap between decentralized training methodologies, Bayesian statistical learning, distributed hypothesis testing, and the computational advantages of variational Bayes' methods.

Related Work: Our fully decentralized training methodology extends federated learning [84, 85, 115] to general graphs in a Bayesian setting and does away with the need of having a centralized controller or IID data distribution across the network. Our learning rule generalizes various Bayesian inference techniques such as [125, 25, 30, 80] and variational continual learning techniques such as [125, 30] to the decentralized learning/training case.

Our work can be viewed as a Bayesian variant of communication-efficient methods based on Stochastic Gradient Descent (SGD) [145, 37, 107] that allow the agents to make several local computations and then periodically average the local models, with the consensus step inspired by previous works on social learning and distributed hypothesis testing [94, 141, 124, 92]. We contrast our work with the vast literature on decentralized optimization based on

SGD [43, 156, 148, 27, 103, 68, 71, 123, 104, 69, 122, 12], where local (stochastic) gradients are computed for each instance of data and communication happens at a rate comparable to number of local updates. The communication rules of decentralized SGD methods originate from prior works on distributed averaging [28, 160], and are either *gossip-based* for doubly stochastic communication matrix [104, 68, 103], or *PushSum-based* [77] for column stochastic communication matrix [122, 12]. In these works typically strong convexity of local objective functions is assumed, with nodes having IID data, and the algorithms and theoretical results are presented in the *realizable* setting, where there exist a parameterization of the model that agrees with the true underlying data generating model. For an overview on the decentralized optimization methods refer to the survey [123].

We also note the relation and difference between our work and decentralized variational Bayesian inference [34, 30]. While these works and ours utilize Bayesian learning agents, [34] merges the local posteriors in a *one-shot* manner after observing all data, and [30] works in a *streaming* setting where data is distributed from a central server and the goal is infer the posterior relevant to the most recent data. Our work combines the advantage of allowing uncertainty from using a Bayesian-like posterior, and the periodic averaging aspect from FedAvg [115]. A recent work [7] fits Gaussian posteriors to local datasets in a federated way, however it is unable to provide theoretical guarantees on convergence rate and can be viewed as a special case to our work when the problem is *realizable* and posterior distribution can shown to remain in the Gaussian family.

Notations: We use boldface for vectors $\mathbf{v}$ and denote its i-th element by v_i. Let $[n] = \{1, 2, \ldots, n\}$. Let $\mathscr{P}(A)$ and $|A|$ denote the set of all probability distributions and the number of elements respectively on a set A. Let $G(\theta, \sigma^2)$ denote the pdf of a Gaussian random variable with mean θ and variance σ^2. Let $D_{\mathrm{KL}}(P_Z||P'_Z)$ be the Kullback–Leibler (KL) divergence between two probability distributions $P_Z, P'_Z \in \mathscr{P}(\mathscr{Z})$.

2.2 The Model: Decentralized Learning

In this section, we formally describe the label generation model at each node, the communication graph, and a criterion for successful learning over the network.

Consider a group of N individual nodes. Each node $i \in [N]$ has access to a dataset $\mathscr{D}^{(i)}$ consisting of instance-label pairs, $\left(X_t^{(i)}, Y_t^{(i)}\right)$ where $i \in [N]$ and $t \geq 1$. Each instance $X_t^{(i)} \in \mathscr{X}_i \subseteq \mathscr{X}$, where $\mathscr{X}_i$ denotes the local instance space of node i and $\mathscr{X}$ denotes a global instance space $\mathscr{X} \subseteq \cup_{i=1}^{N} \mathscr{X}_i$. Similarly, let $\mathscr{Y}$ denote the set of all possible labels over all the nodes[1]. The samples $\left\{X_1^{(i)}, X_2^{(i)}, \ldots, X_t^{(i)}\right\}$ are independent and identically distributed (IID) over time, and are generated according to a distribution $\mathsf{P}_X^{(i)} \in \mathscr{P}(\mathscr{X}_i)$. We view the model generating the labels for each node i as a probabilistic model with a distribution $\mathsf{P}_{Y|X}(y|x)$, $\forall y \in \mathscr{Y}, \forall x \in \mathscr{X}$.

The learners' objective is to (collaboratively) approximate the probabilistic labeling function $\mathsf{P}_{Y|X}(y|x)$ with a parametric probabilistic model $f(\cdot \mid X, \theta)$ with $\theta \in \Theta \subseteq \mathbb{R}^d$ representing the model parameter(s). The next two examples highlight the two complementary and necessary components of learning here: 1) model training utilizing local data, and 2) communication and model aggregation across many agents.

Example 1 (Decentralized Linear Regression)**.** *Consider a linear regression problem [24] with P_X denoting the input distribution over $\mathbb{R}^d$ and latent variable $\theta^* \in \mathbb{R}^d$, which for each input $x \in \mathbb{R}^d$ generates a label $y \in \mathbb{R}$ as $y = \langle \theta^*, x \rangle + \eta$, where $\eta \sim \mathscr{N}(0, \alpha^2)$. Now consider the problem of two agents learning labeling function $\mathsf{P}_{Y|X}(\cdot \mid x) \sim \mathscr{N}(\langle \theta^*, x \rangle, \alpha^2)$ where the agents' local input distributions is restricted to the marginal distributions over sets $\mathscr{X}_1 := \{x \in \mathbb{R}^d : [x_1, \ldots, x_m, 0, \ldots, 0]\}$ and $\mathscr{X}_2 := \{x \in \mathbb{R}^d : [0, \ldots, 0, x_{m+1}, \ldots, x_d]\}$ for some integer m. Let $\mathsf{P}_X^{(1)}$ be the input distribution with support over $\mathscr{X}_1$ and $\mathsf{P}_X^{(2)}$ be the input distribution with support over $\mathscr{X}_2$ and define $\mathsf{P}_X(x) := \mathsf{P}_X^{(1)}(x_{1:m}) \times \mathsf{P}_X^{(2)}(x_{m+1:d})$ for any $x \in \mathbb{R}^d$. It is clear that the observed input distribution for learner 1 lies in $\mathscr{X}_1$, hence learner 1, in isolation, can only learn the first m coordinates of the inputs; similarly,*

[1] Some examples include, $\mathscr{Y} = \mathbb{R}$ for regression and $\mathscr{Y} = \{0, 1\}$ for binary classification.

learner 2 can only access the remaining $d-m$ coordinates locally.

Example 2 (Decentralized Classification with Bayesian Neural Networks)**.** *Consider the problem of training a neural network (NN) with weights $\theta \in \mathbb{R}^d$ and output layer $f(\cdot \mid x, \theta)$ to approximate the true probabilistic labeling function $\mathsf{P}_{Y|X}(\cdot \mid x)$[2] generating labels in $\mathscr{Y}$. Two variational (Bayesian) learners rely on labeled training data $(X_t^{(i)}, Y_t^{(i)})$ which are generated according to a distribution $\mathsf{P}_X^{(i)} \in \mathscr{P}(\mathscr{X}_i)$ to calculate the posterior distribution of the weights $\mathsf{P}(\theta \mid \mathscr{D})$ given the training data*[3]*. Two agents are tasked to arrive at a classifier even when their observations are restricted to non-overlapping partitions of the label space, $\mathscr{Y}_1$ and $\mathscr{Y}_2$ and the input distributions $\mathsf{P}_X^{(1)}$, and $\mathsf{P}_X^{(2)}$.*

In both examples above, neither learners can learn the global modeling in isolation unless there is a method for communication and model aggregation.

Centralized Variational Learning: We first consider the centralized setting as a benchmark in which a (super-)learner is assumed to have access to all agents' training data samples $\mathscr{D} = \{(X_t^{(i)}, Y_t^{(i)}) \in (\mathscr{X}, \mathscr{Y})\}_{i\in[N], t\geq 1}$, where samples are independent and identically distributed (IID) across time with joint distribution $\mathsf{P}_{XY} = \mathsf{P}_X \mathsf{P}_{Y|X}$ and P_X is a measure on $\mathscr{X}$. This centralized learner approximates the true probabilistic labeling function with a parametric probabilistic model with $\theta \in \Theta$.

Towards this objective, a variational (Bayesian) learner places a prior $q^{(0)} = p_\theta \in \mathscr{P}(\Theta)$ on the latent parameter/variable θ and computes the posterior distribution $q^{(t)}(\theta \mid \mathscr{D}) \in \mathscr{P}(\Theta)$ after observing $\mathscr{D}_{1:t} = \{(\mathbf{X}_s, \mathbf{Y}_s)\}_{1\leq s\leq t} = \{(X_s^{(i)}, Y_s^{(i)}) \in (\mathscr{X}, \mathscr{Y})\}_{i\in[N], 1\leq s\leq t}$. These updates follow the variational learner's preference and iterative fashion:

$$q^{(t)} = \underset{\pi\in\mathscr{Q}}{\operatorname{argmin}} \left\{ D_{KL}\left(\pi \,\middle\|\, q^{(t-1)}\right) + \mathbb{E}_\pi\left[-\log f\left(\mathbf{Y}_t \mid \mathbf{X}_t, \cdot\right)\right] \right\}. \tag{2.1}$$

[2]For the case of classification, $f(\cdot \mid x, \theta)$ denotes the final softmax layer, and for the case of regression, $f(\cdot \mid x, \theta)$ is a Gaussian distribution with mean as NN output $f(\cdot \mid x, \theta)$

[3]While exact Bayesian inference on the weights of a neural network is intractable due to large parameter space, in the recent years variational methods to approximate exact Bayesian updates enables Bayesian NN to be learned in a computationally efficient manner [80, 25, 54].

Bernstein–von Mises theorem under model misspecification [82] (See Theorem 4 in Appendix A) asserts that the posterior converges to a point mass at θ^*, explaining the observed labeling distribution:

$$\theta^* \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathsf{P}_X}[D_{KL}(\mathsf{P}_{Y|X}(\cdot \mid X) || f(\cdot \mid X, \theta))] \tag{2.2}$$

Note that given posterior $q^{(t)}(\theta \mid \mathscr{D}_{1:t})$, the predictive distribution of the label of a new test input $\hat{X}$ sampled from P_X is given by

$$\mathsf{P}_t(\cdot \mid \hat{X}) := \int f(\cdot \mid \hat{X}, \theta) dq^{(t)}(\theta \mid \mathscr{D}_{1:t})$$

This means that as the posterior $q^{(t)}(\theta \mid \mathscr{D}_{1:t})$ concentrates on θ^*, the predictive label distribution converges to the best explanation for the observed data.

Definition 1. *A learning problem is said to be realizable if there exists a $\theta^* \in \Theta$ such that $f(\cdot \mid X, \theta^*) = \mathsf{P}_{Y|X}(\cdot \mid X)$ almost surely under P_X. In this case, as shown in Equation* (2.2)*, it is possible to derive the expected loss to zero.*

Decentralized Variational Learning: In this setting, while that the data samples across all learners are labeled by the same unknown probabilistic labeling function $\mathsf{P}_{Y|X}(y \mid x)$ for all $y \in \mathscr{Y}$ given any $x \in \mathscr{X}$, each agent only has access to local data. In other words, each agent $i \in [N]$ approximates this labeling function by $f^{(i)}(\cdot \mid X, \theta)$ for any $X \in \mathscr{X}$. The goal is to agree on a global labeling function f parameterized by $\theta \in \Theta$ which approximates $\mathsf{P}_{Y|X}$ optimally with respect to the global distribution P_X. Formally, the agents are required to collectively optimize the following objective function:

$$\theta^* = \operatorname*{argmin}_{\theta \in \Theta} \sum_{i=1}^{N} \mathbb{E}_{\mathsf{P}_X}[D_{KL}(\mathsf{P}_{Y|X}(\cdot \mid X) || f^{(i)}(\cdot \mid X, \theta))].$$

In this work, we assume variational (Bayesian) agents: agent i places a prior $q_i^{(0)} = p_\theta^{(i)} \in$

$\mathscr{P}(\Theta)$ on the latent variable θ and infers the posterior distribution $q_i^{(t)}(\theta \mid \mathscr{D}_{1:t}^{(i)}) \in \mathscr{P}(\Theta)$ after observing t batches of local data, $\mathscr{D}_{1:t}^{(i)} = (X_s^{(i)}, Y_s^{(i)})_{1 \le s \le t}$, drawn IID from the distribution $\mathsf{P}_{XY}^{(i)}$.

Let us first consider the case of an isolated variational learner whose posterior updates are limited to $\mathscr{D}^{(i)}$. In this setting, agent i's observation is shaped by the local data distribution and might be informationally deficient with respect to the global model. When $\mathsf{P}_{XY}^{(i)} = \mathsf{P}_{XY}$ for all i, then Bernstein-Von Mises Theorem states that the posterior distribution $\mathsf{P}(\theta \mid \mathscr{D}_{1:t}^{(i)})$, where $\mathscr{D}_{1:t}^{(i)} := \{(X_\tau^{(i)}, Y_\tau^{(i)}) : \tau \in [t]\}$, is guaranteed to converge to a globally consistent parameter. In this case, statistically the agents can all learn the global model from local data and there is no need for communication except for boosting the convergence rate.

Definition 2. *We call the local datasets to have non-IID data distributions when there exists $i \in [N]$ for which $\mathsf{P}_{XY}^{(i)} \neq \mathsf{P}_{XY}$. Prior work in federated learning [73, 100, 169] summarizes the following cases of non-IID data distribution:*

1. *label distribution skew:* $\mathsf{P}_Y \neq \mathsf{P}_Y^{(i)}$.
2. *feature distribution skew:* $\mathsf{P}_X \neq \mathsf{P}_X^{(i)}$.
3. *same label different features:* $\mathsf{P}_{X|Y} \neq \mathsf{P}_{X|Y}^{(i)}$.
4. *quantity skew:* $\mathsf{P}_{XY} = \mathsf{P}_{XY}^{(i)}$*, but the number of data samples on each learner is different.*

When the data available to agents locally have non-IID distributions, i.e. $\mathsf{P}_{XY}^{(i)} \neq \mathsf{P}_{XY}$, there is no guarantee that the local models will converge in a globally consistent manner. Therefore, each learner i must aim not only to arrive at a good approximation to its local observations, $\mathscr{D}^{(i)} = \{(X_t^{(i)}, Y_t^{(i)})\}_{t \ge 1}$, but also collaborate with other agents to ensure consenting on a consistent model across. This all suggests that minimizing (2.2) requires communications across learners, which gives rise to our communication model and necessitates our decentralized learning rule.

Before we provide the communications/social network graph model, we note that if $\mathsf{P}_{XY}^{(i)} = \mathsf{P}_{XY}$ for all learners $i \in [N]$, then the distributions of local dataset of all learners are

identical with the global distribution. This case, which is known as federated learning with *IID data distribution*, is not the focus of this work. It is simple to note that if $\mathscr{D}^{(i)} = \{(X_t^{(i)}, Y_t^{(i)})\}_{t\geq 1}$ is identically distributed across nodes according to the joint distribution $\mathsf{P}_{XY}(x,y) = \mathsf{P}_X(x)\mathsf{P}_{Y|X}(y \mid x) = \mathsf{P}_Y(y)\mathsf{P}_{X|Y}(x \mid y)$, then each agent is informationally equivalent to the centralized learner in that the agent is only required to wait longer to essentially receive n times more training data. In other words, in our analysis, we do not consider the problem of data *quantity skew* across the agents [169] since when $\mathsf{P}_{XY} = \mathsf{P}_{XY}^{(i)}$, the difference in the number of data samples on each learner only impacts the rate of learning and not the convergence to θ^* itself which is the focus of this work. We do, on the other hand, consider this phenomenon in our experiments.

Arbitrary Graph for Communication and Model Aggregation: We model the communication between learners via a directed graph with vertex set $[N]$. We define the neighborhood of learner i, denoted by $\mathscr{N}(i)$, as the set of all learners j who have an edge going from j to i. We assume $i \in \mathscr{N}(i)$. Furthermore, if learner $j \in \mathscr{N}(i)$, learner i receives information from learner j. The social interaction of the learners is characterized by a stochastic matrix W, $\sum_{j=1}^N W_{ij} = 1$ and $W_{ii} = 1 - \sum_{j=1, j\neq i}^N W_{ij}$. The weight $W_{ij} \in [0,1]$ is strictly positive if and only if $j \in \mathscr{N}(i)$. The weight W_{ij} denotes the confidence learner i has on the information it receives from learner j.

2.3 Peer-to-peer Variational Learning over an Arbitrary Graph

We now present our peer-to-peer variational learning rule. We assume that at each time instant $t \geq 0$, every learner $i \in [N]$ gets a batch of B observations in the following form: $(\mathbf{X}_t^{(i)}, \mathbf{Y}_t^{(i)}) = (X_{Bt+1:B(t+1)}^{(i)}, Y_{Bt+1:B(t+1)}^{(i)})$. Let Θ denote the latent variable space and typically we choose $\Theta = \mathbb{R}^d$ for some $d > 1$. Each learner takes a Bayesian-like approach and places a prior distribution $p_\theta^{(i)} \in \mathscr{P}(\Theta)$ over the latent variable. Let $q_i^{(t)}$ denote the density of the posterior maintained by each learner i at time t. We introduce a decentralized learning rule which generalizes a learning rule considered in the social learning literature [124, 141, 92], to obtain

the posterior $q_i^{(t)}$ at each learner i at every time instant t. We restrict posterior distributions to a predetermined family of parametric distributions. This allows us to implement the decentralized algorithm in a computationally tractable manner. Let $\mathscr{Q} \subseteq \mathscr{P}(\Theta)$ be a family of posterior parametric distributions.

Each learner i starts with $q_i^{(0)} = p_\theta^{(i)}$ and at every time step $t \geq 0$ the following events happen at every learner $i \in [N]$:

1. Draw a batch of B i.i.d samples $\left(\mathbf{X}_t^{(i)}, \mathbf{Y}_t^{(i)}\right)$ from distribution $\mathsf{P}_{XY}^{(i)}$.

2. **Approximate Bayesian Update Step:** Approximate the local Bayesian update on $q_i^{(t-1)}$ to form a public posterior $b_i^{(t)}$ using the following rule:

$$b_i^{(t)} = \underset{\pi \in \mathscr{Q}}{\operatorname{argmin}} \left\{ D_{KL}\left(\pi \left\| q_i^{(t-1)}\right.\right) + \mathbb{E}_\pi \left[-\log f\left(\mathbf{Y}_t^{(i)} \mid \mathbf{X}_t^{(i)}, \cdot\right)\right] \right\}. \tag{2.3}$$

3. **Communication Step:** Learner i sends $b_i^{(t)}$ to learner j if $i \in \mathscr{N}(j)$ and receives $b_j^{(t)}$ from neighbors $j \in \mathscr{N}(i)$.

4. **Consensus Step:** Update posterior distribution by averaging the log posterior distributions received from neighbors, i.e., for each $\theta \in \Theta$,

$$q_i^{(t)}(\theta) = \frac{\exp\left(\sum_{j=1}^N W_{ij} \log b_j^{(t)}(\theta)\right)}{\int_\Theta \exp\left(\sum_{j=1}^N W_{ij} \log b_j^{(t)}(\phi)\right) d\phi}. \tag{2.4}$$

Remark 2 (Approximate Bayesian update)**.** *Minimization performed in Equation* (2.3) *is referred to as Variational Inference (VI) and the minimand is referred to as the variational free energy or evidence lower bound (ELBO) [121, 25, 80, 54].*

Remark 3 (Gaussian Case)**.** *The variational computation as well as the normalization involved in consensus step (Equation* (2.4)*) require multi-dimensional integration. However, in most practical application Gaussian posterior distributions are used to approximate the true posterior*

and the consensus step reduces to updating the mean and covariance matrices. We will show this in Lemma 1 in Section Sec. 2.4. This also shows that our algoirthm is closely related to the algorithm proposed in [7] where instead of following a consensus step on log posteriors, the authors propose a direct averaging of posteriors.

2.4 Main Results

In this section, we first provide our analytical guarantee for the convergence of our proposed algorithm. Additionally, we provide a special case of our algorithm where the variational posterior class $\mathscr{Q}$ is chosen to be Gaussian. This special case, which later is used in Section 2.5 is shown to reduce the complexity of the algorithm substantially and allow for a closed form characterization of our consensus step.

We now make the following assumptions for the main theorem.

Assumption 1. *The network is a connected aperiodic graph. Specifically, W is an aperiodic and irreducible stochastic matrix.*

Assumption 2. *For all agents $i \in [N]$, assume: (i) The prior $b_i^{(0)}(\theta) > 0$ for all $\theta \in \Theta$. (ii) There exists an $\alpha > 0, L > 0$ such that $\alpha < f^{(i)}(y \mid x, \theta) < L$, for all $y \in \mathscr{Y}$, $\theta \in \Theta$ and $x \in \mathscr{X}$. This guarantees the log-likelihood ratio $\left| \log \frac{f^{(i)}(y|x,\theta)}{f^{(i)}(y|x,\theta')} \right|$ is bounded for all $i \in [N]$, $y \in \mathscr{Y}$, $\theta \in \Theta$ and $x \in \mathscr{X}$.*

Assumption 3. *Let parameter set Φ be a compact subset of $\mathbb{R}^d$, and assume there exists a quantization of Φ with quantization points in Θ such that Θ is an r-covering of Φ. Specifically, we assume there exists a set $\Theta \subset \Phi$ of finite cardinality M that is an r-covering of Φ, i.e. $\Phi \subset \cup_{\theta \in \Theta} \mathscr{B}_r(\theta)$, where $\mathscr{B}_r(\theta) :=$*

$$\left\{ \psi \in \Phi : \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\mathsf{P}_X}[D_{KL}(f^{(i)}(\cdot \mid X, \theta) || f^{(i)}(\cdot \mid X, \psi))] \leq r \right\}$$

Assumption 4. *For all agents $i \in [N]$, there exists an optimal $\theta^* \in \Phi$, such that $f^{(i)}(\cdot \mid X, \theta^*) = \mathsf{P}_{Y|X}(\cdot \mid X)$ almost surely under P_X.*

These assumptions are natural. Assumption 1 states that one can always restrict attention to the connected components of the social network where the information gathered locally by the agents can be disseminated within the component. Assumption 2 prevents the degenerate case where a zero Bayesian prior prohibits learning. Assumption 3 assumes when θ is continuous it belongs to a compact set $\Phi \subset \mathbb{R}^d$. Assumption 4 assume the learning problem is realizable for all agents.

Theorem 1. *Let $\mathscr{Q} = \mathscr{P}(\Theta)$ and $\varepsilon > 0$. Under assumptions 1, 2, 3 and 4, with potentially non-IID local dataset as defined in Definition 2, using the decentralized learning algorithm in Sec. 2.3, with probability at least $1 - \delta$ we have*

$$\max_{i \in [N]} \max_{\theta \in \Theta \setminus \mathscr{B}_r(\theta^*)} b_i^{(T)}(\theta) < e^{-T(K(\Theta)-\varepsilon)}$$

when the number of communication rounds satisfies $T \geq \frac{8C \log \frac{N|\Theta|}{\delta}}{\varepsilon^2(1-\lambda_{max}(W))}$, where we define the rate of convergence of the posterior distribution as follows

$$K(\Theta) := \min_{\theta \in \Theta \setminus \mathscr{B}_r(\theta^*)} \sum_{j=1}^{N} v_j I_j(\theta^*, \theta), \tag{2.5}$$

and $I_j(\theta^*, \theta) := \mathbb{E}_{\mathsf{P}_X^{(j)}}[D_{KL}(\mathsf{P}_{Y|X}(\cdot|\mathbf{X}) || f^{(j)}(\cdot \mid \theta, \mathbf{X})) - D_{KL}(\mathsf{P}_{Y|X}(\cdot|\mathbf{X}) || f^{(j)}(\cdot \mid \theta^*, \mathbf{X}))]$, where eigenvector centrality $\mathbf{v} = [v_1, v_2, \ldots, v_N]$ is the unique stationary distribution of W with strictly positive components, furthermore define $\lambda_{max}(W) := \max_{1 \leq i \leq N-1} \lambda_i(W)$, where $\lambda_i(W)$ denotes i-th eigenvalue of W counted with algebraic multiplicity and $\lambda_0(W) = 1$, and $C := \left|\log \frac{L}{\alpha}\right|$.

Proof of the Theorem 1 is provided in the Appendix A and relies on the following remark.

Remark 4. *If $\mathscr{Q} = \mathscr{P}(\Theta)$, then update performed in equation* (2.3) *reduces to a Bayesian update with prior $q_i^{(t-1)}$ and likelihood function $f^{(i)}\left(\mathbf{Y}_t^{(i)} \mid \mathbf{X}_t^{(i)}, \cdot\right)$, i.e., for $\theta \in \Theta$,*

$$b_i^{(t)}(\theta) = \frac{f^{(i)}\left(\mathbf{Y}_t^{(i)} \mid \mathbf{X}_t^{(i)}, \theta\right) q_i^{(t-1)}(\theta)}{\int_\Theta f^{(i)}\left(\mathbf{Y}_t^{(i)} \mid \mathbf{X}_t^{(i)}, \phi\right) q_i^{(t-1)}(\phi) d\phi}. \tag{2.6}$$

Remark 5. *Proof of Theorem 1 relies on Assumption 4, i.e. the realizable learning problems. We conjecture that this assumption can be relaxed to arrive at a decentralized variant of of Bernstein-Von Mises Theorem. While, this extension remains an interesting future research direction, we investigate relaxing the assumption empirically in Section 2.5 where we consider the non-realizable case of decentralized training of a neural networks over an arbitrary network with non-IID data.*

Remark 6. *Theorem 1 indicates the posterior belief over the parameters θ outside of $\mathscr{B}_r(\theta^*)$ shrinks to zero exponentially fast, with exponent characterized by $K(\Theta)$. The rate of convergence characterized by* (2.5) *is a function of the agent's ability to distinguish between the parameters given by the KL-divergences and structure of the weighted network which is captured by the eigenvector centrality* $\mathbf{v}$ *of the agents. Hence, every agent influences the rate in two ways. Firstly, if the agent has higher eigenvector centrality (i.e. the agent is centrality located), it has larger influence over the posterior distributions of other agents as a result has a greater influence over the rate of exponential decay as well. Secondly, if the agent has high KL-divergence (i.e highly informative local observations that can distinguish between parameters), then again it increases the rate. If an influential agent has highly informative observations then it boosts the rate of convergence. We will illustrate this through extensive simulations in Sec. 2.5.*

Corollary 1 (Average expected loss)**.** *Define $\hat{\theta}_i^{(t)} := \operatorname{argmax}_{\theta \in \Theta} b_i^{(t)}(\theta)$. Let $l(\hat{y}, y)$ denote the loss function of predicting $\hat{y}$ given true label y. The expected loss of agent i under θ is given by $L_i(\theta) = \mathbb{E}_{\mathsf{P}_X}[\int_{\mathscr{Y}} l(\hat{y}, y) f^{(i)}(\hat{y} \mid x, \theta) dy]$. Assume $|l(\hat{y}, y)| \leq B$ for any $\hat{y}, y \in \mathscr{Y}$. Under assumptions 1, 2, 3 and 4, using the decentralized learning algorithm in Sec. 2.3, with probability*

at least $1-\delta$ *we have*

$$\frac{1}{N}\sum_{i=1}^{N}|L_i(\theta^*)-L_i(\hat{\theta}_i^{(T)})|\leq\frac{B\sqrt{r}}{2}$$

When the number of communication rounds satisfies $T\geq\frac{8C\log\frac{N|\Theta|}{\delta}}{\varepsilon^2(1-\lambda_{max}(W))}$.

Proof of Corollary 1 is provided in the Appendix A.

Corollary 2. *Let* $\Theta\subset\mathbb{R}^d$ *be a finite set, i.e.* $\Theta=\{\theta_1,...,\theta_M\}$. *Let* $\mathscr{Q}=\mathscr{P}(\Theta)$ *and* $\varepsilon>0$. *Under assumptions 1, 2, and assume for all agents* $i\in[N]$, *there exists an optimal* $\theta^*\in\Theta$, *such that* $f^{(i)}(\cdot\mid X,\theta^*)=\mathsf{P}_{Y|X}(\cdot\mid X)$ *almost surely under* P_X. *Then using the decentralized learning algorithm in Sec. 2.3, with probability at least* $1-\delta$ *we have*

$$\max_{i\in[N]}\max_{\theta\neq\theta^*}b_i^{(T)}(\theta)<e^{-T(K(\Theta)-\varepsilon)}$$

when $T\geq\frac{8C\log\frac{NM}{\delta}}{\varepsilon^2(1-\lambda_{max}(W))}$, *and the rate of convergence is:* $K(\Theta)=\min_{\theta\neq\theta^*}\sum_{j=1}^{N}v_jI_j(\theta^*,\theta)$.

At the end of this section, we discuss a special case of our algorithm where in both variational training and consensus step, we restrict our attention to the family of Gaussian distributions. Let $\mathscr{Q}$ denote the family of Gaussian posterior distributions with pdf given by $G(\mu,\Sigma)$.

Here we first note that If $\mathscr{Q}\subsetneq\mathscr{P}(\Theta)$, equation (2.3), finds a distribution $\pi\in\mathscr{Q}$ which satisfies the following

$$\underset{\pi\in\mathscr{Q}}{\operatorname{argmin}}D_{KL}\left(\pi\middle\|\frac{f^{(i)}\left(\mathbf{Y}_t^{(i)}\mid\mathbf{X}_t^{(i)},\theta\right)q_i^{(t-1)}(\theta)}{\int_\Theta f^{(i)}\left(\mathbf{Y}_t^{(i)}\mid\mathbf{X}_t^{(i)},\phi\right)q_i^{(t-1)}(\phi)d\phi}\right). \tag{2.7}$$

In other words, equation (2.7) projects the distribution obtained via equation (2.6) onto the allowed family of posterior distributions $\mathscr{Q}$ by employing KL-divergence minimization.

Lemma 1 (Posterior merging with Gaussians)**.** *Let* $(\mu_i^{(t)}, \Sigma_i^{(t)})$ *denote the mean and the covariance matrix of* $b_i^{(t)}$ *at learner i. Then the posterior distribution* $q_i^{(t)}$ *obtained after the consensus step also belongs to* $\mathcal{Q}$*. Furthermore, the closed-form update of the mean and covariance matrix* $(\widetilde{\mu}_i^{(t)}, \widetilde{\Sigma}_i^{(t)})$ *of* $q_i^{(t)}$ *is given as follows*

$$(\widetilde{\Sigma}_i^{(t)})^{-1} = \sum_{j=1}^{N} W_{ij} {\Sigma_j^{(t)}}^{-1}, \quad \widetilde{\mu}_i^{(t)} = \widetilde{\Sigma}_i^{(t)} \sum_{j=1}^{N} W_{ij} (\Sigma_j^{(t)})^{-1} \mu_j^{(t)}. \tag{2.8}$$

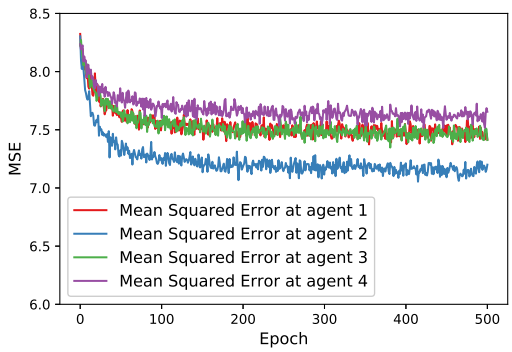
Derivation of equation (2.8) is provided in the Appendix A.

2.5 Experiments

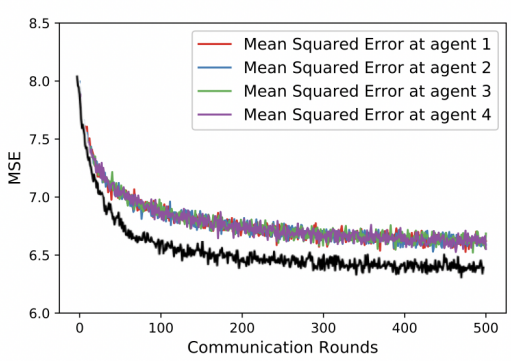
We construct experiments in the decentralized Bayesian linear regression setup (Example 1) and decentralized classification setup (Example 2) over synthetic and real-world benchmark datasets. We demonstrate the performance of the proposed decentralized learning rule in all cases of non-IID data distributions mentioned in Definition 2. Furthermore, we discuss the effect of eigenvector centrality on the rate of convergence, compare our learning rule with FedAvg [115] over a federated dataset, and show how our method scales with larger number of agents in a time-varying network.

2.5.1 Decentralized Bayesian Linear Regression

To illustrate our approach, we construct an example of Bayesian linear regression (Example 1) in the realizable setting over the network with 4 agents. Let $\theta^* = [-0.3, 0.5, 0.5, 0.1]^T$ and let noise be distributed as $\eta \sim \mathcal{N}(0, \alpha^2)$ where $\alpha = 0.8$. Agent i makes observations $(\mathbf{x}, y)$, where $\mathbf{x} = [0 \ldots, 0, x_i, 0, \ldots, 0]^T$ and x_i is sampled from Unif$[-1, 1]$ for $i = 1$, Unif$[-1.5, 1.5]$ for $i = 2$, Unif$[-1.25, 1.25]$ for $i = 3$, and Unif$[-0.75, 0.75]$ for $i = 4$. We assume each agent starts with a Gaussian prior over θ with zero mean vector and covariance matrix given by diag$[0.5, 0.5, 0.5, 0.5]$, where diag$(\mathbf{x})$ denotes a diagonal matrix with diagonal elements given by vector $\mathbf{x}$. The social interaction weights are given as $\mathbf{W}_1 = [0.5, 0.5, 0, 0]$, $\mathbf{W}_2 = [0.3, 0.1, 0.3, 0.3]$,



(a) Learning without cooperation



(b) Learning with cooperation

Figure 2.1. Mean Squared Error (MSE) of the predictions over a test dataset under two cases: (i) all agents, despite the severe deficiency of their observations, learn without cooperation using local training data only, and (ii) agents learn using the proposed decentralized learning rule. The black curve represents a benchmark scenario a central agent learns the model with access to all coordinates of the training data.

$\mathbf{W}_3 = [0, 0.5, 0.5, 0]$ and $\mathbf{W}_4 = [0, 0.5, 0, 0.5]$. Since each agent starts with a Gaussian prior over Θ, the posterior distribution after a Bayesian update remains Gaussian, which implies $\mathscr{Q}$ remains fixed as the family of Gaussian distributions and the consensus step reduces to equation (2.8).

We show that our proposed social learning framework enables a fully decentralized and fast learning of a global model even when the local data is severely deficient. More specifically, we assume that each agent makes observations along only one coordinate of $\mathbf{x}$ even though the global test set consists of observations belonging to any $\mathbf{x}$. Note that this is a case of extreme non-IID data partition across the agents and corresponds to case (2) of *non-IID* in Definition 2. Fig. 2.1b shows that the MSE of all agents, when trained using the decentralized learning rule, is much lower than training separately with no communication, and matches that of a central agent, implying that the agents converge to the true θ^* as our theory predicts.

2.5.2 Decentralized Image Classification

To illustrate the performance of our learning rule on real datasets we consider the problem of decentralized training of Bayesian neural networks for an image classification task on the

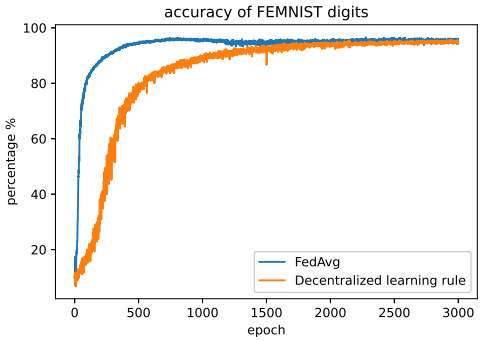
MNIST digits dataset [99], the Fashion-MNIST (FMNIST) dataset [159] and the Federated Extended-MNIST dataset from LEAF [33]. For all our experiments we consider multilayered Bayesian NN and employ Monte Carlo to obtain the predictions.

We divide the training dataset into non-overlapping subsets. Hence, agents must learn $\mathbf{b}_i^{(n)}$ such that the resulting predictive distribution can perform well over the global dataset without sharing the local data and hence not having seen input example associated with the labels that are missing locally.

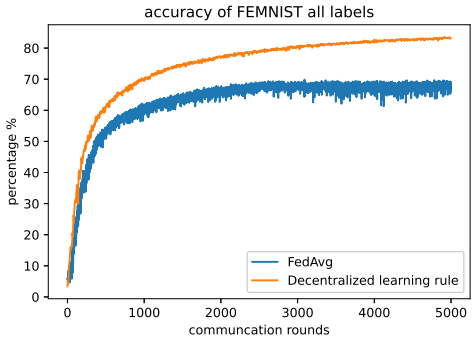
Comparison to FedAvg on federated dataset

In this section, we compare the proposed decentralized update rule with Federated Averaging (FedAvg) [115] on a public federated dataset called Federated Extended-MNIST (FEMNIST) [33]. The Federated Extended-MNIST (FEMNIST) dataset is part of LEAF, a benchmark dataset for federated learning [33]. The dataset has handwritten digits and lower/upper case alphabets by different writers, therefore there is a total of 62 labels, namely $[0,...,9]$ and $[a,...,z,A,...,Z]$. The dataset has a total of 3550 writers with an average of 226 images per writer, where the images are of size 28 by 28. In the experiment setup, each learner only sees digits/alphabets from a single writer, therefore there is significant *feature skew* (case (2) of *non-IID* in Definition 2).

For this experiment we use a multi-layer Convolutional Neural Network. The input image is first passed through a convolution layer with 6 channels, size 5 by 5, followed by a 2DMaxpool of size 2 by 2 and a ReLU. Then the output goes through another convolution layer with 16 channels, size 5 by 5, followed by a 2DMaxpool of size 2 by 2 and a ReLU. Then the output is forwarded to 2 linear layers with 784 and 120 hidden units each. We choose $\mathscr{Q}$ to be the family of Gaussian mean-field approximate posterior distributions with pdf given by $G(\theta,\mu,\Sigma)$, where Σ is a strictly diagonal matrix [25, 125]. As discussed in Remark. 4 this corresponds to performing variational inference to obtain a Gaussian approximation of the local posterior distribution, i.e., minimizing the variational free energy given in equation (2.3) over $\mathscr{Q}$. While



(a) On digits only



(b) On digits and upper/lower case alphabets

Figure 2.2. Figure shows average accuracies over all nodes of proposed decentralized learning algorithm and FedAvg on FEMNIST dataset.

we compute the KL divergence in (2.3) in a closed form, we employ simple Monte Carlo to compute the gradients using Bayes by Backprop [25, 80].

To emulate the federated learning setup, where the beliefs of all learners are merged with equal weight to form a public belief, we use a densely connected network with $W_{ij} = 1/N$ for all $i, j \in [N]$. We note that FedAvg can also be viewed as a decentralized SGD method over a fully connected network. We use a total of 50 learners, chosen randomly from the 3550 available writers. We use local training batch size $B = 10$, local epochs $E = 1$ and a total of 3000 communication rounds for digits only and 5000 communication rounds for digits and alphabets. For all agents, we use Adam optimizer [79] with initial learning rate of 0.001 and learning rate decay of 0.99 per communication round.

On only digits, our method is on par with FedAvg, with accuracy $\sim 96\%$. On both digits and alphabets, our method significantly outperforms FedAvg with accuracy over 83% after 5000 communication rounds, comparing to $\sim 69\%$ from FedAvg. We note that classifying both digits/alphabets is significantly harder than just digits, and by putting a prior on the latent parameters our method encodes uncertainty in the local estimates of the true parameter and thus prevents premature convergence to local data, as is suffered by FedAvg.

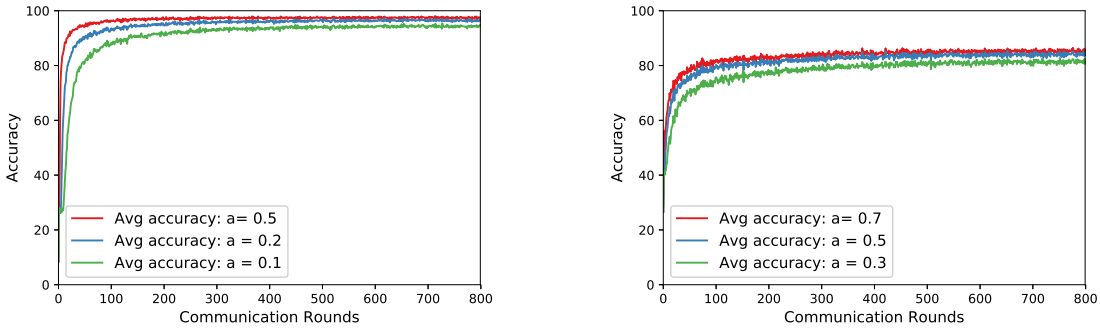
Effect of eigenvector centrality

In this section, we investigate how the eigenvector centrality of an agent affect the rate of convergence to the true parameter. We examine this on a network with a star topology, where a central agent is connected to 8 other edge agents. Let the social interaction weights for the central agent be $\mathbf{W}_1 = [1/9, \ldots, 1/9]$. For $a \in (0,1)$, we assume that an edge agent i puts a confidence $\mathbf{W}_{i1} = a$ on the central agent, $\mathbf{W}_{ii} = 1-a$ on itself and zero on others. Note that as the confidence a which the edge agents put on the central agent increases, the eigenvector centrality of the central agent v_1 increases i.e., central agent becomes more influential over the network.

We consider two datasets: (i) the MNIST digits dataset [99] where each image is assigned a label in $\mathscr{Y} = [0, \ldots, 9]$ and (ii) the Fashion-MNIST (FMNIST) dataset [159] where each image is assigned a label in $\mathscr{Y} = [$ `t-shirt`, `trouser`, `pullover`, `dress`, `coat`, `sandal`, `shirt`, `sneaker`, `bag`, `ankle-boot`$]$. Both datasets consist of 60,000 training images and 10,000 testing images of size 28 by 28. For all our experiments we consider a fully connected NN with 2-hidden layers with 200 units each using ReLU activations which is same as the architecture considered in the context of federated learning in [115]. Again as in Section 2.5.2, we choose $\mathscr{Q}$ to be the family of Gaussian mean-field approximate posterior distributions.

We partition the dataset such that the central agent has more informative local observations. Specifically, the central agent sees more labels and also has the largest number of samples, which corresponds to *label skew* (case (1)) and *quantity skew* (case (4)) of *non-IID* data distribution in Definition 2.

We vary confidence a which the edge agents put on the central agent over $[0.1, 0.2, 0.3, 0.5, 0.7]$, and therefore the eigenvector centrality of the central agent v_1 increases as $[0.1, 0.18, 0.25, 0.36, 0.44]$. We partition the MNIST dataset into two subsets so that the central agent dataset has all images of labels $[2, \ldots, 9]$ and edge agents has all images of labels $[0, 1]$. To ensure all the edge agents has equal number of images, we shuffle the images with labels $[0, 1]$ and partition them into 8 non-overlapping subsets. Similarly, for Fashion-MNIST (FMNIST)



(a) On MNIST dataset **(b)** On FMNIST dataset

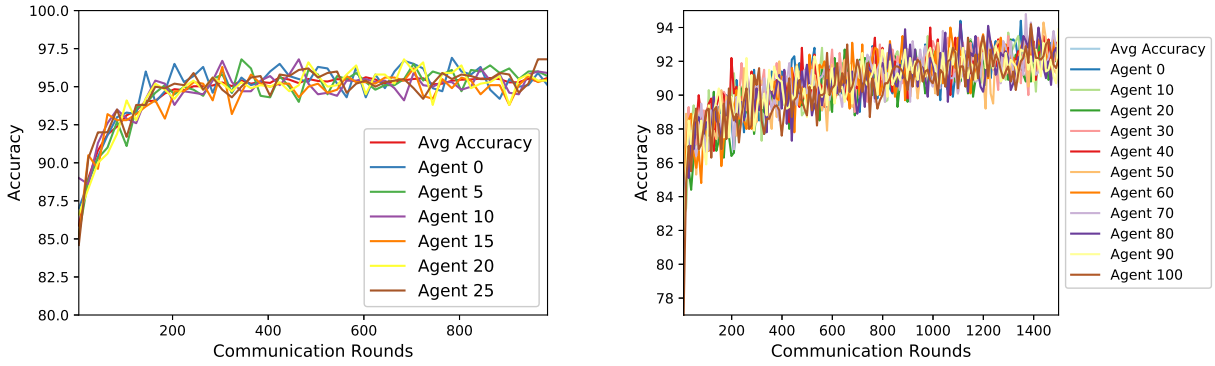
Figure 2.3. Figure shows the variation in the average accuracy over a star network topology as the eigenvector centrality of the central agent is varied.

dataset, we first partition into two subsets so that central agent has access to labels [`t-shirt`, `pullover`, `dress`, `coat`, `shirt`, `bag`] and edge agents have access to labels [`trouser`, `sandal`, `sneaker`, `ankle-boot`]. We shuffle the images with labels [`trouser`, `sandal`, `sneaker`, `ankle-boot`] and partition them into 8 non-overlapping subsets.

We ensure that all agents has same number of local updates u per communication round, which is equal to $(\lfloor n_{edge}/B \rfloor)E$, here n_{edge} denotes the number of training samples for each edge node. For the central agent, this means that for each local epoch, the central agent is trained on a random subset of its local dataset, whereas the edge agents use all the local dataset. For all agents, we use Adam optimizer [79] with initial learning rate of 0.001 and learning rate decay of 0.99 per communication round.

Table 2.1. Settings for Star Topology Network Experiment: E is number of local epochs, B is the local minibatch size, u is the number of local updates per communication round, η is the initial learning rate for all agents, ε is the learning rate decay rate, n_{center} is the dataset size of the central agent, n_{edge} is the dataset size of each of the edge agent.

Experiment	E	B	u	η	ε	n_{center}	n_{edge}	comm rounds
`MNIST`	5	50	155	0.001	0.99	47335	1583	800
`FMNIST`	5	100	150	0.001	0.99	36000	3000	800



(a) Average accuracy over all 26 nodes.

(b) Accuracy over all 100 nodes.

Figure 2.4. Figure shows the accuracies of selected agents in a time-varying network.

From equation (2.5) we know that placing more confidence a on the central agent increases the rate of convergence to the true parameter and increases rate of convergence of the test dataset accuracy. This is demonstrated in Fig. 2.3a and Fig. 2.3b where both accuracy and the rate of convergence improve as a increases. In other words, rate of convergence and the average accuracy is the highest when the agent with most informative local observations has most influence on the network. Note that assigning too much confidence on the most informative agents can potentially hinder other nodes from learning from their local dataset.

Asynchronous Decentralized Learning on Time-varying Networks

Now we implement our learning rule on time-varying networks which model practical peer-to-peer networks where synchronous updates are not easy or very costly to implement. We consider a time-varying network of $N+1$ agents numbered as $\{0,1,\dots,N\}$. At any give time, only N_0 agents are connected to agent 0 in a star topology. For $k \in [N/N_0]$, let $\mathcal{G}_k$ denote a graph with a star topology where the central agent 0 is connected to edge agents whose indices belong to $\{N_0(k-1)+1,\dots,N_0k\}$. This implies at any given time only a small fraction of agents N_0/N are training over their local data. Note that $\cup_{k=1}^{N/N_0}\mathcal{G}_k$ is strongly connected network over all $N+1$ agents. The social interaction weights for the central agent are $\mathbf{W}_0 = [1/N_0+1,\dots,1/N_0+1]$. Let

$a = 0.5$. An edge agent $i \in \mathcal{G}_k$ puts a confidence $\mathbf{W}_{i0} = a$ on the central agent 0, $\mathbf{W}_{ii} = 1 - a$ on itself and zero on others. The MNIST dataset is divided in an i.i.d manner, i.e., data is shuffled and each agent is randomly assigned approximately $(60{,}000/N+1)$ samples. As demonstrated in Fig. 2.4, for $N = 25, N_0 = 5$, we obtain an average accuracy of 96.5% over all agents and 95.1% accuracy at the central agent and for $N = 100, N_0 = 10$, we obtain an average accuracy of 92.3% over all agents and 93.1% accuracy at the central agent. This also demonstrates that decentralized learning can be achieved with as few as 600 samples locally.

Table 2.2. Settings for Time-varying Network Experiment: E is number of local epochs, B is the local minibatch size, u is the number of local updates per communication round, η is the initial learning rate for all agents, ε is the learning rate decay rate, n is the dataset size of any agent. Since all agents have same number of samples, they automatically have equal number of local updates per communication round. Adam optimizer is used for all agents.

Experiment	E	B	u	η	ε	n	comm rounds
`N = 25`	1	50	47	0.001	0.99	2307	1000
`N = 100`	2	10	120	0.001	0.998	594	1000

2.6 Conclusion and Future Work

In this chapter, we consider the problem of peer-to-peer decentralized variational learning over an arbitrary social network, which is an extension of the classical Federated Learning setup by allowing peer-to-peer communications. Building on prior work on distributed hypothesis testing [94] and variational Bayesian inference [25, 30, 80], we propose a fully decentralized variational learning algorithm consisting of two main components of variational updating based on local data and log-posterior averaging. In the realizable setting, we show this method to converge to the correct global labeling function and provide an analytical and closed-form characterization of the rate of convergence. Empirically, we validate our theoretical finding, illustrate the advantages over existing methods such as federated averaging when the data is

non-IID across the agents. We also illustrate how the choice of the underlying social network impacts the rate of convergence.

2.7 Acknowledgements

Chapter 2, in full, is a reprint of the material published in the IEEE Journal on Selected Areas in Information Theory (JSAIT), vol. 3, no. 2, pp. 172-182, June 2022: "Peer-to-Peer Variational Federated Learning over Arbitrary Graphs" (Xinghan Wang, Anusha Lalitha, Tara Javidi, and Farinaz Koushanfar). The dissertation author was the primary investigator and author of this paper. This work was supported by NSF-CNS under award number 2016737.

Chapter 3

Contextual Shortest Path: a Centralized RL problem

We have familiarized ourselves with the Federated Learning portion of Multi-Agent Reinforcement Learning in Chapter 2. Now we turn our attention to Reinforcement Learning, specifically, a centralized one: a single agent interacts with the unknown stochastic environment, while attempting to make better decisions given limited feedback information.

In this chapter, we study a new class of stochastic decision problems, which we call the Contextual Shortest Path (CSP) problems. The problem is motivated by dynamic path planning and obstacle avoidance for UAV and drone applications. More specifically, in path planning applications, there is a need to compute the path between two pre-determined locations in space while avoiding various spontaneous obstacles in the environment. In general, finding the optimal path in a stochastic environment with time-varying obstacles generalizes the problem of finding the shortest path over an undirected graph with stochastic edges, where random edge realizations are augmented with richer contexts that can change over time and require online inspection.

Under perfect knowledge of context distributions, we provide an extended Dijkstra's algorithm to solve the associated dynamic program efficiently. When the context distributions are unknown and need to be learned online, we first adapt two algorithms as our baselines, one based on Thompson Sampling and the other based on ε-greedy exploration. We then propose a novel reinforcement learning algorithm, RL-CSP, which intelligently distributes exploration

episodes over the time horizon and ensures the agent visits under-explored states. We bound the regret for RL-CSP, and augment the theoretical results with simulations over various network topologies. We further demonstrate the improved robustness of our RL-based algorithm in the stochastic shortest path setting.

3.1 Introduction

Classic shortest path algorithms aim to find the optimal path between a pair of source and destination vertices in a directed or undirected graph. The optimal path is defined to be the path with the least total cost when edge costs are fixed, [41, 21, 48], and the path with the least expected cost when edge costs are drawn from unknown distributions, [53, 110, 18, 170]. However, in many applications, such as autonomous vehicle navigation, the optimal path is not fixed, but depends on contextual information which may be received at any time. For example, a drone navigating towards a destination may observe an obstacle using its on-board sensors, this additional context could force the drone to take an alternative path. Naively, the agent can re-plan the optimal path using classic shortest path algorithms each time a new context is observed, but this is costly and makes no use of past experiences. However, if the agent has learned that obstacles appear with high frequency at a given location, it can more efficiently plan by avoiding that location on future interactions with the environment. Our goal, therefore, is to solve the shortest path problem in the presence of time-varying contexts while also learning the environment.

The Contextual Shortest Path problem can be cast as a multi-armed bandit (MAB) problem. In the MAB formulation, at each time step an agent selects one of M independent arms and incurs a random cost drawn i.i.d. from an unknown distribution for that arm. The goal of the agent is to minimize cumulative regret over horizon T, relative to an omniscient player who always picks the optimal arm. It has been shown by [89] that cumulative regret grows at least logarithmically with time. If the arms are independent of each other, then each arm

must be explored directly as they offer no information about each other, so optimal regret grows linearly with the number of arms. We identify our contextual shortest path problem with the MAB formulation by treating each contextual path selection policy as an arm. However, since there exist $M! \sim O(M^M)$ possible distinct policies for a MDP with M states, the optimal regret grows at least exponentially with the number of states. So, directly applying any standard MAB algorithm by rank ordering all possible contextual path selection policies results in excessive regret. In this paper, we propose a learning-based algorithm that leverages local information about the states and transitions within the model to more efficiently learn optimal behavior. This allows better cumulative regret scaling in both the number of states and contexts.

Contributions: Our contribution is threefold. First, we formulate a new class of Markov decision problems which we call *Contextual Shortest Path* (CSP). We then provide an extended Dijkstra's algorithm to solve the associated dynamic program when the distributions of contexts are known. Second, we adapt two standard algorithms, based on Thompson Sampling and ε-greedy exploration, to solve the CSP problem without knowledge of the context distributions. We then propose a novel reinforcement learning algorithm, RL-CSP, which intelligently distributes exploration episodes over the time horizon, with sparser exploration as the agent learns its environment, and ensures that the agent visits under-explored states. We further prove that the cumulative regret of RL-CSP is polynomial in both the number of states and contexts and is logarithmic in time. Lastly, we empirically evaluate the performance of our RL-CSP algorithm and compare it to the baseline provided by our adapted algorithms. We run experiments on three different graph topologies designed to exhibit challenging instances of CSP problems. We report the cumulative regret for all three algorithms, demonstrating that our RL-CSP algorithm more quickly converges to optimal behavior and accumulates lower regret than either of the baseline algorithms. We also adapt RL-CSP to the Stochastic Shortest Path (SSP) problem to demonstrate its improved robustness relative to the baseline Thompson Sampling algorithm when edge costs are less distinguishable.

Related Work: This work is related to Stochastic Shortest Path (SSP), episodic Markov

Decision Process, and Contextual Markov Decision Process (CMDP).

1) Stochastic Shortest Path (SSP): SSP problems as widely studied in [53, 110, 18, 170] generally assume the transition cost of each edge is drawn i.i.d from some probability distribution. In the absence of statistical knowledge, the agent chooses the optimal route base on the current estimation of these probability distributions. Our CSP problem can be reduced to a SSP problem with Bernoulli distributed transition costs if acquisition actions are fixed and each state, under any given context, only has a single available neighbor.

2) Episodic Markov Decision Process: Our CSP formulation can be viewed as a special case of an extended episodic MDP [64, 172, 137], where each episode in episodic MDP is the random interval from origin state to termination state. Despite minor difference between CSP and episodic MDP, we benchmark our proposed algorithms against Thompson Sampling [64]. We note that our CSP problem does not satisfy the loop-free [172] or the fully connected [137] assumptions, and [172, 137] cannot be readily applied in our setting.

3) Contextual Markov Decision Process (CMDP): Our proposed CSP problem is inherently distinct from a Contextual Markov Decision Process (CMDP). The CMDP formulation [117, 56] is an episodic MDP in which the context is fully revealed before each episode. CMDP then maps the context to an unknown MDP from a finite set of candidate MDPs and, within an episode, the transition probabilities of the MDP are fixed. In our setting, a context is acquired prior to each state transition and that context then governs the transition probabilities. In other words, the optimal policy has to account the observed context at each time step during an episode.

3.2 Problem formulation

We consider a type of episodic Markov Decision Process (MDP), which we refer to as the Contextual Shortest Path (CSP). Let $G = (\mathscr{V}, \mathscr{E})$ be an undirected graph. Let $s \in \mathscr{V}$ be source state and $d \in \mathscr{V}$ be destination state. Let T be the total number of episodes, where each episode $t = 1, 2, ..., T$ begins from state s at $n = 0$ and ends at $n = \tau_t = \min\{\tau_{max}, \tau_{d,t}\}$, where

$\tau_{d,t}$ is the time the agent reaches d in episode t and τ_{max} is maximum episode duration. Let $i_{n,t}$ denote the current state at timestep n of episode t. The agent then takes context acquiring action $a_{n,t} \in \mathscr{A}$, which reveals context $X_{n,t}$. Specifically, given state i and context acquiring action a, the observed context $X \in \mathscr{X}_i$ is drawn from the probability distribution $\mathbf{P}_i^a := P(\cdot|i,a)$, where $\mathscr{X}_i$ is the finite space of contexts observable in state i. The agent then incurs context acquisition cost $c_A(i_{n,t}, a_{n,t})$.

After the agent observes context $X_{n,t}$ in state $i_{n,t}$, it takes transition action $u_{n,t} \in \mathscr{N}_{X_{n,t}}(i_{n,t})$, where $\mathscr{N}_{X_{n,t}}(i_{n,t})$ denotes available neighbors for $i_{n,t}$, including $i_{n,t}$ itself, under context $X_{n,t}$. Note that the context may constrain which neighbors the agent can transition to. The agent then reaches next state $i_{n+1,t} = u_{n,t}$ and incurs transition cost $c_U(i_{n,t}, X_{n,t}, u_{n,t})$. The reward r_t for episode t is R if the agent reaches d before τ_{max} ($\tau_{d,t} \leq \tau_{max}$), and 0 otherwise.

The goal of the CSP problem is to find a sequence of actions $a_{n,t}$ and $u_{n,t}$ for $n = 0, 1, ..., \tau_t$ and $t = 1, 2, ..., T$, such that the expected reward J_T is maximized:

$$J_T = \mathbb{E}\left[\sum_{t=1}^{T}\{r_t - \sum_{n=0}^{\tau_t}[c_U(i_{n,t}, X_{n,t}, u_{n,t}) + c_A(i_{n,t}, a_{n,t})]\}\right]$$

Here the expectation is taken over context realizations, context acquisition actions and state transition actions. If J^* denotes the optimal expected reward under perfect knowledge of the model, then maximizing total expected reward is equivalent to minimize the regret defined as: $R_\pi(T) = TJ^* - J_T$.

3.3 Algorithms for Contextual Shortest Path

3.3.1 Known context distributions $\mathbf{P}_i^a$

We propose a Dijkstra-type algorithm that gives the optimal context acquiring actions and state transition actions for all states and possible contexts, when the context distributions $\mathbf{P}_i^a$ are fully known. Define the value function $V^* : \mathscr{V} \to \mathbb{R}$ by the following system of equations:

$$V^*(d) = R$$

$$V^*(i) = \max\left\{0, \max_{a\in\mathscr{A}}\left\{-c_A(i,a)+ \sum_{X\in\mathscr{X}_i} P(X|i,a) \max_{j\in\mathscr{N}_X(i)}\{(V^*(j)-c_U(i,X,j)\}\right\}\right\}$$

for $i \neq d$

This value function rewards states for their proximity to the destination, while penalizing them for the associated context acquisition and transition costs. We propose a Dijkstra-type algorithm, EXTENDED-DIJKSTRA-CSP to solve for the value functions. For any subset of vertices, $\mathscr{C} \subseteq \mathscr{V}$, define the associated frontier of vertices accessible from $\mathscr{C}$

$$\mathscr{N}(\mathscr{C}) := \{j : \exists i \in \mathscr{C}, s.t. P(X|i,a) > 0, X \in \mathscr{X}_i, j \in \mathscr{N}_X(i)\}$$

Our EXTENDED-DIJKSTRA-CSP algorithm solves for the optimal V^* and in doing so finds the optimal context acquisition action $a^*(i)$ for each $i \in \mathscr{V}$. Then, for state i, given context $X \in \mathscr{X}_i$, the optimal transition action is $u^*(i) = \text{argmax}_{j\in\mathscr{N}_X(i)}\{V^*(j) - c_U(i,X,j)\}$.

3.3.2 Unknown context distributions $\mathbf{P}_i^a$

In the case where contexts are drawn from unknown distributions, $\mathbf{P}_i^a$, we first present two baseline model-based algorithms: an ε-greedy approach and a Thompson Sampling based approach. We then present our novel algorithm, RL-CSP, which adaptively distributes exploration and exploitation episodes and in exploration episodes, optimizes for information acquisition by its choice of context acquisition actions and by prioritizing under-explored states.

ε-greedy-CSP

We first present a model-based ε-greedy algorithm for the CSP problem, which we call ε-greedy-CSP. We maintain counting variable $N(i,X,a)$ which tracks the number of times context

Algorithm 1. EXTENDED-DIJKSTRA-CSP. Inputs: $G = (\mathscr{V}, \mathscr{E})$, $\mathbf{P}_i^a$, c_U, c_A, source s, destination d. Outputs: $V^*(i)$ and $a^*(i)$ for all $i \in \mathscr{V}$.

Initialize $\mathscr{O} = d$, $V^*(d) = R$, $\mathscr{A} = \mathscr{V} - d$
while $\mathscr{A} \neq \emptyset$ **do**
  for each $i \in \mathscr{A} \cap \mathscr{N}(\mathscr{O})$ and $X \in \mathscr{X}_i$ **do**
    $W(X,i) = \max_{u \in \mathscr{O} \cap \mathscr{N}_X(i)} \{-c_U(i,X,u) + V^*(u)\}$
    $\overline{\mathscr{X}}(Y,i) := \{X \in \mathscr{X}_i : W(X,i) \geq W(Y,i)\}$
    if $W(X,i) < 0$ for all X **then**
      $V(i) = 0$
    else {set $\mathscr{X}_0(i) = \{X : W(X,i) \geq 0\}$}
      $V(i) = \max_{a, Y \in \mathscr{X}_0(i)} \dfrac{-c_A(i,a) + \sum_{X \in \overline{\mathscr{X}}(Y,i)} P(X|i,a) W(X,i)}{\sum_{X \in \overline{\mathscr{X}}(Y,i)} P(X|i,a)}$
      $a(i), Y(i) = \operatorname{argmax}_{a, Y \in \mathscr{X}_0(i)} \dfrac{-c_A(i,a) + \sum_{X \in \overline{\mathscr{X}}(Y,i)} P(X|i,a) W(X,i)}{\sum_{X \in \overline{\mathscr{X}}(Y,i)} P(X|i,a)}$
    end if
  end for
  $l = \operatorname{argmax}_{i \in \mathscr{A}} V(i)$
  Set $\mathscr{O} = \mathscr{O} \cup \{l\}$ and $\mathscr{A} = \mathscr{A} - \{l\}$
  Set $V^*(l) = V(l)$ and $a^*(l) = a(l)$
end while

X is observed in state i, after taking context acquiring action a. We use these counts to maintain an empirical estimate of the context distributions, $\hat{P}(X|i,a) = N(i,X,a)/\sum_{X\in\mathscr{X}_i} N(i,X,a)$. $\hat{P}(X|i,a)$ is used to update optimal context acquiring action $\hat{a}(\cdot)$ and value function $\hat{V}(\cdot)$ by EXTENDED-DIJKSTRA-CSP at the end of each episode. For each episode, when the agent is in state i, it takes context acquiring action $\hat{a}(i)$, observes context X, and with probability $(1-\varepsilon(i,X))$ where $\varepsilon(i,X) = \frac{1}{1+\sum_{a\in\mathscr{A}} N(i,X,a)}$, takes greedy transition action (exploitation) $j = \text{argmax}_{j\in\mathscr{N}_X(i)} \hat{V}(j) - c_U(i,X,j)$. Otherwise, a uniform random action is taken (exploration).

Thompson Sampling

We also adapt Thompson Sampling, [139, 64], to the CSP problem. For an episodic MDP defined by the tuple $(\mathscr{S},\mathscr{A},P,R,H)$, where $\mathscr{S}$ is the state space, $\mathscr{A}$ is the action space, $P = P(\cdot|s,a)$ are the unknown transition probabilities, $R = R(\cdot|s,a)$ is the unknown stage cost distribution and H is the horizon, Thompson Sampling algorithms put prior distributions on the transition and cost distributions. For each episode t, P_t and R_t are sampled from the current distribution over P and R. This fixes an MDP defined by $(\mathscr{S},\mathscr{A},P_t,R_t,H)$ for which an optimal policy is calculated. Then, given the transitions and costs observed by the agent under that policy, a Bayesian update is performed on the distributions over P and R.

Here, for the CSP problem, our unknowns are the context distributions, $\mathbf{P}_i^a := P(X|i,a)$. Since $\mathscr{X}_i$ is finite, we put a Dirichlet prior on $\mathbf{P}_i^a$ with $|\mathscr{X}_i|$ parameters. The Dirichlet distribution is the conjugate prior for multinomial distributions and so allows for a simple, closed form posterior update. All parameters of each of the Dirichlets are initialized to 1, but we examine in our experiments the strong dependence the performance of this algorithm has on this choice of prior.

RL-CSP with Least Traversed Node

We now propose a reinforcement-learning based algorithm, RL-CSP, with distinct exploration and exploitation episodes. In exploration episodes, the agent is routed through the

least visited state via a path that prioritizes visiting under-explored states along the way. Whereas in exploitation episodes, the agent takes greedy actions based on $\hat{V}(\cdot)$ and $\hat{a}(\cdot)$ calculated under the same running estimate of context distributions, $\hat{P}(X|i,a)$, as in ε-greedy-CSP. We ensure that at every episode, t we have performed $g \log t$ exploration episodes, where g is a constant that depends on the size of the context spaces and how distinguishable values of different states are This bound guarantees optimal regret scaling in both time and the size of the problem while ensures that the algorithm will explore more frequently in early episodes and become greedier as it learns its environment.

In the exploration episodes we employ the following scheme: for each state $i \in \mathscr{V}$, let $N(i)$ be number of times i is visited and let $i^* = \text{argmin}_j N(j)$ be the least visited state. Let $\mathscr{P}$ be set of all paths from s to d, and let $\mathscr{P} \ni p = (s, v_1, v_2, ..., d)$ be one of the paths. The exploration route p^* is taken to be the minimizer $p^* = \text{argmin}_{\{p \in \mathscr{P}: i^* \in p\}} \sum_{j \in p} N(j)$. That is, the path which visits state i^* and minimizes the total number of times that each state along the path has already been visited. This can be solved by Dijkstra's shortest path algorithm with the counts as the costs.

Theorem 1 (Regret bound for RL-CSP) *Let B be the maximum cardinality of any of the context spaces, i.e. $|\mathscr{X}_i| \leq B$, $\forall i \in \mathscr{V}$. Let $\Delta = \min_{i \in \mathscr{V}} \{\min_{j \in \mathscr{N}(i)} |V^*(i) - V^*(j)|\}$. Now if we let $g = \frac{N\alpha^2 B^2}{2\Delta^2}$, where $\alpha = R \sum_{i=1}^{|\mathscr{V}|} \sum_u \sum_a \frac{1}{c_U(i,X,u) + c_A(i,a)}$ and R is the reward for reaching state d and let $M = |\mathscr{V}|$, then the algorithm* RL-CSP *achieves regret $\mathscr{O}(Mg \log(T))$.*

Proof See Appendix B for complete proof.

3.4 Experiments

We use stochastic graphs as a test-bed for these three algorithms, using each to solve shortest path problems both with and without context on a variety of graph topologies. These topologies, displayed in Figure 3.1, are chosen to illustrate problem instances where the ε-greedy and Thompson sampling based algorithms struggle, and to demonstrate the practical convergence

Algorithm 2. RL-CSP. Inputs: $G = (\mathscr{V}, \mathscr{E})$, constant g, number of exploration phases N_{exp}. Initialize $N(i)$,$N(i,X,a)$ to 0. Initialize $\hat{V}(i) = 0$ for $i \neq r$, and $\hat{V}(r) = R$. Initialize $\hat{a}(i)$ randomly. p_k^* denotes the k^{th} state in the route p^*.

for $t = 1$ **to** T **do**
  if $N_{exp} < g\log(t)$ **then**
    $p^* = \text{argmin}_{p \in \mathscr{P} \cap i \in p} \sum_{j \in p} N(j)$
    for $k = 0$ to $p^*.length() - 1$ **do**
      Take $\hat{a}(p_k^*)$, observe X, $N(p_k^*, X, \hat{a}(p_k^*))++$
      while $p_{k+1}^* \notin \mathscr{N}_X(i)$ **do**
        $u(p_k^*) = p_k^*$, $N(p_k^*)++$
        Take $\hat{a}(p_k^*)$, observe X, $N(p_k^*, X, \hat{a}(p_k^*))++$
      end while
      $u(p_k^*) = p_{k+1}^*$, $N(p_{k+1}^*)++$
    end for
    Update $\hat{P}(X|i,a) = N(i,X,a) / \sum_{X \in \mathscr{X}_i} N(i,X,a)$
    Obtain $\hat{V}(\cdot)$, $\hat{a}(\cdot)$ by EXTENDED-DIJKSTRA-CSP
    $N_{exp}++$
  else
    Take greedy actions based on using $\hat{V}(\cdot)$, $\hat{a}(\cdot)$
    For all (i,X,a) triplets observed on path:
    $N(i)++$, $N(i,X,a)++$
    Update $\hat{P}(X|i,a) = N(i,X,a) / \sum_{X \in \mathscr{X}_i} N(i,X,a)$
  end if
end for

improvements offered by our RL-CSP algorithm.

1) Contextual Shortest Path (CSP): Our first simulations apply each of the three presented algorithms to Contextual Shortest Path problems. We consider a graph with stochastic edges, which are only available with some probability $p \in [0,1]$ at each time step, and an agent which must learn the edge probabilities while solving the shortest path problem between vertices s and d. The context acquisition actions here are to observe some subset of the agent's current neighbor vertices and see which are available — the agent is unable to transition to a state without first observing that it is currently available. Let $i_{n,t}$ denote current state at timestep n and episode t. The agent acquires context $X_{n,t} \in \{0,1\}^{|\mathcal{N}(i_{n,t})|}$, which indicates the current available neighbors from state $i_{n,t}$. For any state $i \in V$, let $\mathbf{P}_i^a = P(X|i,a) = P([X_1, X_2, ..., X_{|\mathcal{N}(i)|}]|i,a)$ denote the conditional probability distribution of observing context $X \in \{0,1\}^{|\mathcal{N}(i)|}$. If all edges are independent from each other, then

$$P([X_1, ..., X_{|\mathcal{N}(i)|}]|i,a) = \prod_{k=1}^{|\mathcal{N}(i)|} P(X_k|i,a) = \prod_{k=1}^{|\mathcal{N}(i)|} p_{i,k}^a$$

where $p_{i,k}^a$ is the probability that the edge between states i and k exists given that context acquisition action a was taken.

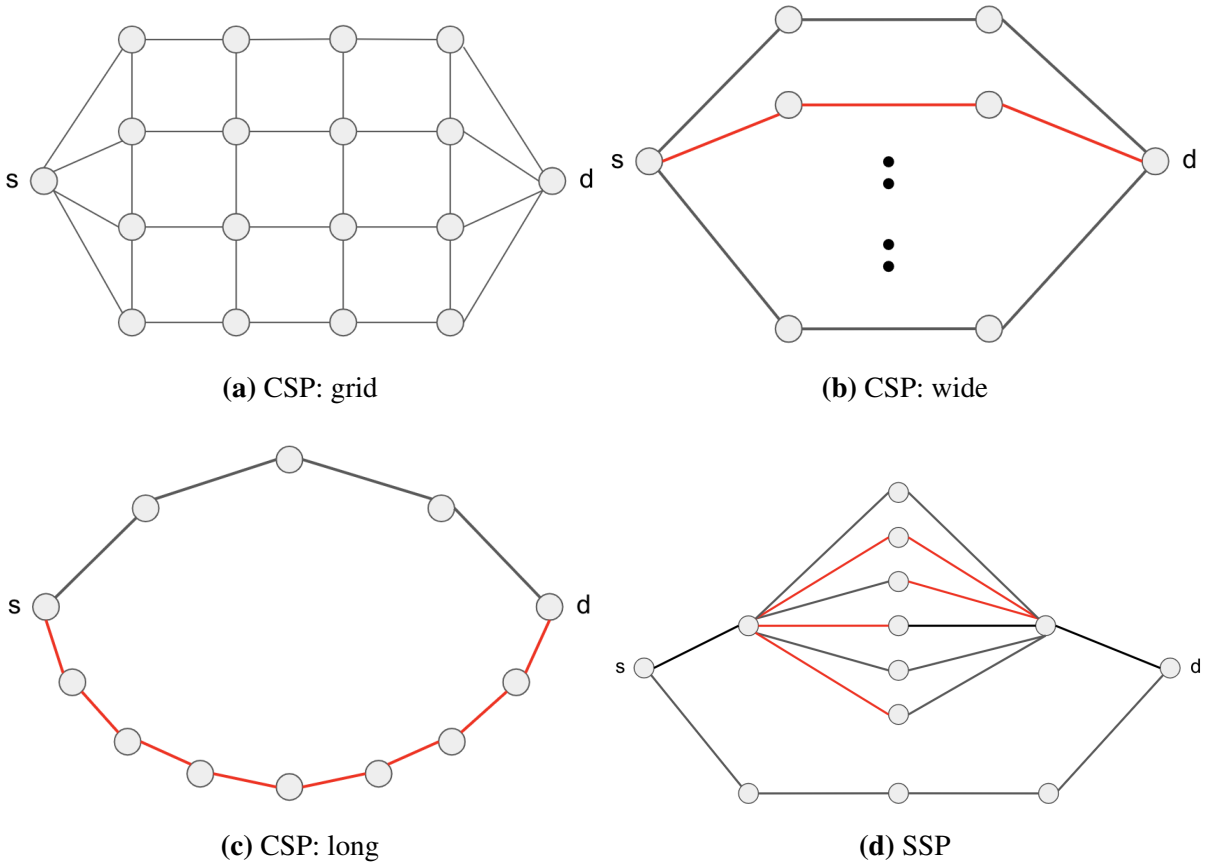
For simplicity, we assume equal transition costs $c_U(i,X,j)$. We also assume that transition costs c_U and context acquisition costs c_A are known. This is realistic since these quantities will generally be known parameters of the agent, e.g. the energy cost to run the sensors in a drone.

Setup: For the three graphs considered, the edges in the underlying model are independent for simplicity of demonstration.

- The grid network (Figure.3.1(a)) has a 4-by-4 grid between s and d. For all the edges (i,j), $p_{i,j} = 0.7$. There can be more than one optimal route at a time.
- The wide network (Figure.3.1(b)) has 8 parallel 3-hop routes from s to d. For all the links (i,j), $p_{i,j} = 0.3$, except for the middle link (k,l) of the red route which has $p_{k,l} = 0.9$,

which makes it the optimal route.

- The long network (Figure.3.1(c)) has two routes from s to d. For link (i, j) on the upper, sub-optimal route, $p_{i,j} = 0.4$. For link (k, l) on the lower, optimal route, $p_{k,l} = 0.9$.



(a) CSP: grid

(b) CSP: wide

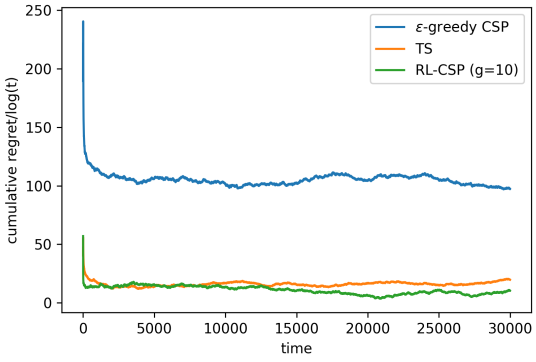
(c) CSP: long

(d) SSP

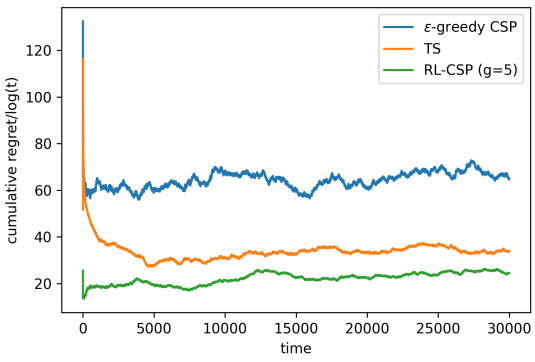
Figure 3.1. Network topologies. s denotes source and d denotes destination. For (b) and (c), the red route is the expected optimal route. For (d) the red edges are those with smaller expected weights.

Results: Figure 3.2 displays the cumulative regret, averaged over 50 trials for each of the three algorithms over the different topologies shown in Figure 3.1(a)(b)(c).

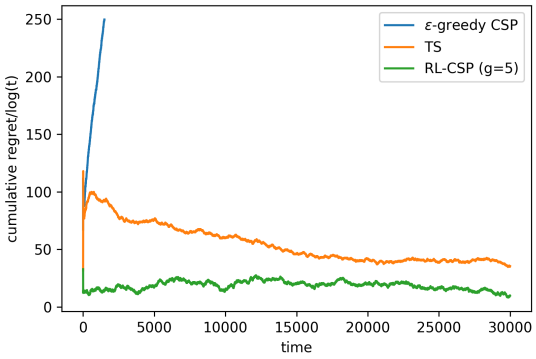
Figure.3.2(a) shows that all three algorithms achieve optimal asymptotic regret order of $\mathcal{O}(\log t)$. Here RL-CSP and Thompson Sampling have comparable performance, while the ε-greedy algorithm incurs significant additional regret over the early episodes as its unguided exploration learns the network much more slowly.



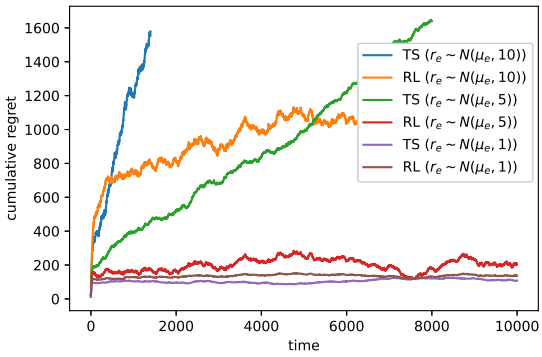
(a) $R(t)/log(t)$ for grid network



(b) $R(t)/log(t)$ for wide network



(c) $R(t)/log(t)$ for long network



(d) $R(t)$ for SSP

Figure 3.2. (**a**)(**b**)(**c**) Regret under different network topologies for CSP. (**d**) Regret for SSP.

Figures 3.2(b) and (c) demonstrate the practical shortcomings of Thompson Sampling on certain graph topologies. The wide network in (b) and the long network in (c) demonstrate two instances where Thompson Sampling under-performs our RL-CSP algorithm. This occurs for two primary reasons: first, Thompson Sampling is very sensitive to the choice of prior, and in the case where the prior is not chosen concentrated near the true value of the parameters, it may require many updates to converge to accurate estimates of the model. And second, Thompson Sampling has no way of prioritizing high-information but low-reward exploration and so it may never take costly actions that facilitate quickly learning the environment. In contrast, RL-CSP requires no specification of any prior distribution or detailed model and, in its exploration episodes, prioritizes information acquisition over cost. These advantages are evident in the faster regret convergence displayed in Figure 3.2(b),(c). We also note that the value of g in Theorem 2, which governs number of exploration episodes, is a conservative choice to guarantee our theoretical convergence results. In the simulations we are able to achieve $\mathscr{O}(\log T)$ regret with a much smaller choice of g.

2) Stochastic Shortest Path (SSP): We note that the idea from RL-CSP can be applied to the widely studied Stochastic Shortest Path (SSP) problem [53, 110, 18, 170]. Consider an undirected graph $G = (\mathscr{V}, \mathscr{E})$ with m edges and n vertices. Let $\mathscr{P}$ be set of all simple (loop-free) paths from s to r. At episode t, a m dimensional binary vector $a(t) \in \mathscr{P}$ is chosen, where $a_i(t) = 1$ means the i^{th} edge is on the path. At time t, the cost $W_i(t)$ of for each edge $i \in \mathscr{E}$ is drawn i.i.d. from a distribution with unknown mean, $\mathbb{E}[W_i(t)] = \theta_i$. Here, we use a normal distribution, $W_i(t) \sim \mathscr{N}(\theta_i, \tilde{\sigma}^2)$, where $\tilde{\sigma}$ is known. The total cost of of path $a(t)$ is: $C(t) = \sum_{i=1}^{m} a_i(t) W_i(t)$ and the regret of a policy π is: $R_\pi(T) = \mathbb{E}_\pi[\sum_{t=1}^{T} C(t)] - TC^*$, where C^* is the expected cost of optimal path.

Here we present RL-SSP, which is an adaptation of RL-CSP to the SSP problem. RL-SSP proceeds by taking forced exploration steps with diminishing frequency, each of which selects paths that include the least visited states in the graph. And we compare this to TS-SSP, which applies Thompson Sampling as in [139] by placing $\mathscr{N}(\mu_i, \sigma_i^2)$ priors on the unknown

Algorithm 3. RL-SSP($\mathscr{P}, g, T$). Initialize $(\hat{\theta})_{1\times m}$ (empirical mean of edge cost), $(k)_{1\times m}$ (No. of visits to edges), $N_{exp} = 0$.

for $t = 1$ **to** T **do**
  if $N_{exp} < g\log(t)$ **then**
    Take $p^* = \underset{p\in\mathscr{P}\cap i\in p}{\operatorname{argmin}} \sum_{j\in p} k_j$, $i = \underset{j}{\operatorname{argmin}} k_j$
    Update $k_i, \hat{\theta}_i$ from $W_i(t)$ for edges i on p^*, $N_{exp}++$
  else
    $p = \operatorname{argmin}_{p\in\mathscr{P}} \sum_{i\in\mathscr{E}} p_i\hat{\theta}_i$
    Update $k_i, \hat{\theta}_i$ from $W_i(t)$ for edges i on p
  end if
end for

Algorithm 4. TS-SSP ($\mathscr{P}, \tilde{\sigma}^2, T$). Initialize (μ_i, σ_i^2) for $i \in \mathscr{E}$.

for $t = 1$ **to** T **do**
  sample $\hat{\theta}_i \sim \mathscr{N}(\mu_i, \sigma_i^2)$
  Take $p = \underset{p\in\mathscr{P}}{\operatorname{argmin}} \sum_{i\in\mathscr{E}} p_i\hat{\theta}_i$, observe $W_i(t)$ for $i \in p$
  for $i \in p$ **do**
    $(\mu_i, \sigma_i^2) \leftarrow \left(\frac{\frac{1}{\sigma_i^2}\mu_i + \frac{1}{\tilde{\sigma}^2}W_i(t)}{\frac{1}{\sigma_i^2}+\frac{1}{\tilde{\sigma}^2}}, \frac{1}{\frac{1}{\sigma_i^2}+\frac{1}{\tilde{\sigma}^2}}\right)$
  end for
end for

edge means θ_i.

Results: To compare these algorithms and further examine the influence of TS' priors, we use the topology shown in Figure.3.1(d). The edges in red have mean $\theta_i = 10$ while the remaining edges have $\theta_i = 20$, which creates a single optimal path through the top branch alongside many sub-optimal paths. We consider a range of variances for the edge costs, $\tilde{\sigma} \in \{1, 5, 10\}$ and observe the cumulative regret for both TL-SSP and RL-SSP for each value of $\tilde{\sigma}$ in Figure.3.2(d). We observe that TS-SSP's performance is on par with RL-SSP when $\tilde{\sigma}$ is small, since in that case it is relatively easy to distinguish the optimal route from others. However, as $\tilde{\sigma}$ increases, TS-SSP incurs nearly linear regret. This occurs when Thompson Sampling has a mis-specified prior and is unable to explore sufficiently to compensate, and so the algorithm converges to a sub-optimal path. We note that under-exploration caused by an inaccurate prior is also addressed by [131]. Furthermore, if the underlying distribution family of θ_i is unknown, a poorly chosen prior for Thompson Sampling based algorithm can also lead to poor regret [60]. Our RL-SSP algorithm avoids these pitfalls as it guarantees sufficient exploration of each edge and does not require any specification of a prior distribution or detailed model.

3.5 Conclusion

In this chapter, the Contextual Shortest Path problem, a generalization of the standard graph shortest path problem that allows for the inclusion of contexts as side information, is considered. We propose a Dynamic Programming based algorithm that is optimal when the conditional distribution of contexts is known. Under unknown context distributions, we propose two algorithms, ADAPT-CSP, based on Q-learning, and RL-CSP which trades off exploration and exploitation. We prove the time-averaged regret of ADAPT-CSP converges to zero and the cumulative regret of RL-CSP is logarithmic in time and polynomial in the size of the graph. Simulations on different network topologies confirms our theoretical results, and empirically demonstrates that our algorithm outperforms Thompson Sampling in some scenarios for both

CSP and SSP problems.

3.6 Acknowledgements

Chapter 3, in full, is a reprint of the material published in the 55th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, pp. 471-476, 2021: "Contextual Shortest Path with Unknown Context Distributions" (Xinghan Wang, Gregory Fields, and Tara Javidi). The dissertation author was the primary investigator and author of this paper. This work was partially supported by ARO under award number W911NF-19-1-0317.

Chapter 4

Federated Certainty Equivalence Control: a Decentralized RL problem

In this chapter, we study a class of Multi-Agent Reinforcement Learning (MARL) problem, namely the decentralized linear quadratic (LQ) control problem. In this scenario, the agents are facing a linear Gaussian system with unknown transition dynamics where agents have partial state observations and can control partial inputs over an additive channel. We propose an algorithm that strikes a balance between system identification (exploration) and certainty equivalence control (exploitation) in order to minimize regret and communication cost. Our non-asymptotic analysis demonstrates that regret of our algorithm scales at a rate of $\mathcal{O}(\sqrt{T})$ for a time horizon of T, while maintaining low communication between agents. Numerical analysis provides validation for our regret analysis and facilitates comparisons between various exploration strategies.

4.1 Introduction

Decentralized multi-agent systems are ubiquitous across various applications such as decentralized control of robots and drones [8, 5], decentralized autonomous vehicles [153], non-cooperative games [50], among others. Extensive research in the literature has focused on decentralized multi-agent systems with known system dynamics, exploring various frameworks such as decentralized optimal control [119, 158, 164], multi-agent planning [128], and non-

cooperative game theory [58]. However, realistically, the environment model is often only partially observed or entirely unknown. Multi-agent reinforcement Learning (MARL) [165] is designed to address the broader context of multi-agent sequential decision-making, where the agents lack complete knowledge of the environment model. In such conditions, agents learn the environment by interacting with the system and gathering rewards.

In this study, we focus on decentralized LQ problem, which is a multi-agent learning problem with linear system dynamics and quadratic cost function. In particular, we consider a scenario with two agents, each of which observes the system state partially. The system is controlled by the collective actions of the two agents, i.e., the sum of the actions. Leveraging LQ systems as a learning benchmark holds considerable advantages due to their theoretical tractability and extensive relevance across diverse engineering domains. Our particular problem arises for instance when robots that are decoupled in their dynamics and observations are tasked with collaborating to have coupled behavior.

Reinforcement Learning (RL) and MARL can be broadly categorized into two approaches: model-based and model-free. In model-free approaches [63], the policy is directly optimized by interacting with the environment and reward collection. Conversely, model-based approaches [118] involve learning the environment model through interaction with the system and subsequently determining the optimal policy based on the estimated system model.

The model-based approaches for decentralized LQ problem are very limited. In [11, 10] the authors consider a decentralized multi-agent system with nested information structure. Our work compliments this line of work by considering a more general information structure. In the absence of nested information structure, there needs to be carefully designed collaboration strategies in place for the agents to learn the system collaboratively.

In this paper, we propose FedCE, a collaborative learning policy which allows agents to efficiently explore and exploit. In our proposed algorithm, we partition time into exploitation and exploration intervals, carefully designing their durations to achieve high performance in both system model learning and minimizing regret. During the exploration phase, we employ

Least Square Estimation (LSE) techniques to obtain local partial system model estimates. These estimates are then shared between agents at the end of each exploration interval. Subsequently, both agents compute Certainty Equivalence controllers, which they apply during the exploitation interval. As time progresses and the model estimates improve, the relative length of exploration intervals compared to exploitation intervals decreases, leading to reduced communication between agents over time. We analyze FedCE in terms of its regret bound and demonstrate that the regret scales at a rate of $\mathcal{O}(\sqrt{T})$ for a time horizon of T.

In summary, our contributions in this paper are as follows.

- We propose FedCE, the first federated (decentralized yet collaborative) algorithm consisting of three building blocks; (1) Certainty equivalence controller for exploitation, (2) Coordinated exploration, and (3) Communication and knowledge sharing.
- We show that it is possible to schedule the algorithm components such that the rate of communication and control policy computation is negligible.
- We show that the regret in FedCE scales at a rate of $\mathcal{O}(\sqrt{T})$ for a time horizon of T.
- We provide results confirming the value of collaboration between the two agents.
- We provide extensive numerical analysis that support our theoretical results.

The remainder of the paper is structured as follows. In Section 4.2, we review the problem of linear quadratic (LQ) control with known system model and the associated optimal linear controller. In Section 4.3, we formally introduce the problem of decentralized LQ control with unknown system model. We then propose our adaptive algorithm for the considered problem in Section 4.4, which we call FedCE. In Section 4.5, we analyze the learning of FedCE algorithm, as well as the growth rate of the regret. We conclude with extensive simulations in Section 4.6 that validate our theoretical analysis.

4.1.1 Notation

We use lowercase and uppercase letters for vectors and matrices, respectively. We denote the i-th element of vector v by v_i. We use $||\cdot||$ to denote the Euclidean norm for vectors and matrices. $\mathcal{N}(\theta,\sigma)$ denotes the pdf of a Gaussian random variable with mean θ and variance σ^2. We denote the trace of matrix A by $tr(A)$. The identity matrix of size $n\times n$ is denoted by I_n.

4.2 Preliminaries

In this section, we provide some background on the well-known problem of centralized LQ control as the baseline of the problem that we have studied in this paper and present the known results in this area.

4.2.1 Centralized LQ control with known system model

Consider a discrete-time linear system with the following system dynamics:

$$x(t+1) = Ax(t) + Bu(t) + w(t),$$

where $x(t)\in\mathbb{R}^n$ is the state of the system, $u(t)\in\mathbb{R}^d$ is the control input from some controller, and $w(t)\in\mathbb{R}^n$ is the random disturbance which is sampled i.i.d. from a Gaussian distribution $\mathcal{N}(0,\sigma_w I_n)$. $A\in\mathbb{R}^{n\times n}$ and $B\in\mathbb{R}^{n\times d}$ are matrices representing the system model. We denote the system model by $\theta = [A,B]^T$.

The step cost incurred at time t, given current state $x(t)$ and input $u(t)$, is a quadratic function defined as

$$c(t) = x(t)^T Q x(t) + u(t)^T R u(t). \tag{4.1}$$

The goal is to find a control policy $\pi:\mathbb{R}^n\to\mathbb{R}^d$, that minimizes the infinite-horizon average expected cost:

$$J^\pi(\theta) = \limsup_{T\to\infty} \frac{1}{T}\mathbb{E}^\pi[\sum_{t=1}^{T} c(t)]$$

It is common to make the following assumptions to solve the above problem.

Assumption 5. *The cost matrices Q and R are symmetric and positive definite. Without loss of generality, we assume that the minimum singular value of R is greater than or equal to 1.*

Assumption 6. *The system dynamics (A,B) is stabilizable, i.e. there exists K such that $(A+BK)$ is stable.*

When θ is known, given Assumptions 5 and 6, we have a well-studied stochastic Linear Quadratic Control problem, where it is known that the optimal controller is a linear feedback controller (Linear Quadratic Regulator, or LQR) of the form: $u(t) = -K(\theta)x(t)$, where $K(\theta)$ is given by

$$K(\theta) = (R + B^T P(\theta) B)^{-1} B^T P(\theta) A, \tag{4.2}$$

and P^* is the solution to the Algebraic Riccati Equation:

$$\begin{aligned} P(\theta) = Q + A^T P(\theta) A \\ - A^T P(\theta) B (R + B^T P(\theta) B)^{-1} B^T P(\theta) A. \end{aligned} \tag{4.3}$$

We also know that the optimal cost under this LQR is:

$$J^*(\theta) = \sigma_w^2 tr(P(\theta))$$

4.2.2 Centralized Adaptive LQ control with unknown system model

When the system model $\theta = [A,B]^T$ is unknown, we are facing an online setting where the system dynamics need to be learned as the agent interacts with the system. The performance of a policy π is measured by the cumulative regret in horizon T, which is defined as:

$$\mathscr{R}(\pi, T) = \sum_{t=1}^{T} \Big[c(t) - J^*(\theta) \Big]$$

which is the difference between the cost incurred by π and the optimal infinite horizon averaged cost. The goal is to find adaptive policy π that minimizes the cumulative regret.

4.3 Problem Statement: Decentralized LQ control with unknown system model

In this paper, we study the decentralized version of the LQ problem stated in section 4.2 with unknown system model. Consider a discrete-time linear system jointly controlled by N agents, each having a partial observation of the system state. We define $\mathscr{N} = \{1, \cdots, N\}$ to be the set of all agents. The agents are distributed over an undirected graph $\mathscr{G} = (\mathscr{N}, \mathscr{E})$, over which they can communicate with each other. The system dynamics is given by

$$\begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ \cdots \\ x_N(t+1) \end{bmatrix} = \begin{bmatrix} A_{11} & 0 & \cdots & 0 \\ 0 & A_{22} & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & A_{NN} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ \cdots \\ x_N(t) \end{bmatrix} + \begin{bmatrix} B_{11} & 0 & \cdots & 0 \\ 0 & B_{22} & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & B_{NN} \end{bmatrix} \Big(\sum_{i=1}^{N} u_i(t) \Big) + \begin{bmatrix} w_1(t) \\ w_2(t) \\ \cdots \\ w_N(t) \end{bmatrix}$$

Each agent i observes a partial state $x_i(t) \in \mathbb{R}^{n_i}$, with $\sum_{i=1}^{N} n_i = n$ and $A^{ii} \in \mathbb{R}^{n_i \times n_i}$ for $i \in \mathscr{N}$. Furthermore, each agent i is associated with a local control input $u_i(t) \in \mathbb{R}^d$, and $B^{ii} \in \mathbb{R}^{n_i \times d}$. The joint control input to the system is $u(t) = \sum_{i=1}^{N} u_i(t)$. We assume the additive noise is sampled i.i.d. from a Gaussian distribution $\mathscr{N}(0, \sigma_w I_n)$. We also assume that the system model matrices are unknown to both of the agents.

Let $x(t) = \begin{bmatrix} x_1(t) \\ \cdots \\ x_N(t) \end{bmatrix}$, $w(t) = \begin{bmatrix} w_1(t) \\ \cdots \\ w_N(t) \end{bmatrix}$, and

$$A^* = \begin{bmatrix} A_{11} & \cdots & 0 \\ & \ddots & \\ 0 & \cdots & A_{NN} \end{bmatrix}, B^* = \begin{bmatrix} B_{11} & \cdots & 0 \\ & \ddots & \\ 0 & \cdots & B_{NN} \end{bmatrix},$$

then, the system dynamics can be rewritten as

$$x(t+1) = A^* x(t) + B^* u(t) + w(t).$$

Given state $x(t)$ and joint control $u(t)$, the agents receive a quadratic cost of

$$c(t) = x(t)^T Q x(t) + u(t)^T R u(t) \tag{4.4}$$

We assume the above problem satisfies assumptions 5, 6.

The above cost function can represent a variety of costs that we often see in real-world scenarios. The following corollary presents some of the examples of the well-known objectives that can be modeled by the above quadratic cost function.

Corollary 3. *The following scenarios are some examples with an objectives that can be modeled with a quadratic cost function $c(t) = x(t)^T Q x(t) + u(t)^T R u(t)$:*

- ***Zero Target Control**: This is the simplest case that can be modeled by the quadratic cost function of $c(t) = x(t)^T Q x(t) + u(t)^T R u(t)$, where $x(t)$ is the state that is targeted to reach zero.*

- ***Waypoint Target Control**: This control scenario corresponds to a case where the target of the system state is a waypoint, x^*. This objective can be initially represented by the cost function $c(t) = (x(t) - x^*)^T Q (x(t) - x^*) + u(t)^T R u(t)$. However, to present this cost function in the form of a simple quadratic cost function, one can augment the system state $x(t)$ with x^* to create constant and linear terms as follows. If we define $\hat{x}(t) = \begin{bmatrix} x(t) \\ x^* \end{bmatrix}$, then by defining $\hat{Q} = \begin{bmatrix} Q, -Q \\ -Q, Q \end{bmatrix}$, the cost function $c(t) = \hat{x(t)}^T \hat{Q} \hat{x(t)} + u(t)^T R u(t)$ will be equal to the desired cost function.*

- ***Route Tracking Control****: This scenario captures the cases where there is a predetermined path, $x^*(t)$ that the state $x(t)$ needs to follow. To fall within the linear model assumption of this paper, we assume that $x^*(t)$ also has a linear dynamic as follows.*

$$x^*(t+1) = Hx^*(t),$$

where the parameter H is known to all agents.

To mimic such path following scenario, we set the cost function to be $c(t) = (x(t) - x^*(t))^T Q(x(t) - x^*(t)) + u(t)^T Ru(t)$. To fit this cost function into the format of equation (4.9)*, we define the augmented state $\hat{x} = \begin{bmatrix} x(t) \\ x^*(t) \end{bmatrix}$. Then, the system dynamic parameters would be $\hat{A} = \begin{bmatrix} A & 0 \\ 0 & H \end{bmatrix}$, and $\hat{B} = \begin{bmatrix} B & 0 \\ 0 & 0 \end{bmatrix}$. We also define $\hat{Q} = \begin{bmatrix} Q, -Q \\ -Q, Q \end{bmatrix}$. Then, we can write $c(t) = \hat{x}(t)^T \hat{Q} \hat{x}(t) + u(t)^T Ru(t)$.*

Given the decentralized nature of the problem, the goal is to find control policies $\pi(t) = (\pi_1(t), \pi_2(t))$, where $\pi_i(t) : \mathbb{R}^{n_i} \to \mathbb{R}^d$ and $u_i(t) = \pi_i(t)[x_i(t)]$, that minimizes the infinite-horizon average expected cost

$$J^\pi(\theta) = \limsup_{T \to \infty} \frac{1}{T} \mathbb{E}^\pi [\sum_{t=1}^{T} c(t)]$$

In the rest of the paper, we will refer to the above problem as the DecLQ problem.

In the next lemma, we present the optimal controllers when the system model $\theta^* = [A^*, B^*]^T$ is known. The structure of this controller will later be used for the case when the system model is not known.

Lemma 2. *In DecLQ problem, if the system model $\theta^* = [A^*, B^*]^T$ is known, the optimal controller is given by*

$$u_i(t) = \begin{bmatrix} K_{1i}(\theta^*) \\ \dots \\ K_{Ni}(\theta^*) \end{bmatrix} x_i(t),$$

where

$$K(\theta^*) = \begin{bmatrix} K_{11}(\theta^*) & \cdots & K_{1N}(\theta^*) \\ & \ddots & \\ K_{N1}(\theta^*) & \cdots & K_{NN}(\theta^*) \end{bmatrix}$$

is calculated from equations (4.2) *and* (4.3)*. We also have the optimal cost to be* $J^*(\theta^*)$*, which is the optimal cost of the centralized controller.*

As mentioned before, the structure of the optimal control for the case where the system model is known will be used in the algorithm that we propose for the the case where the system model is unknown. If θ^* is unknown, the agents need to learn it in order to minimize the infinite horizon average cost. The performance of the learners is measured by the cumulative regret over horizon T,

$$\mathscr{R}(\pi, T) = \sum_{t=1}^{T} \Big[c(t) - J^*(\theta^*) \Big]$$

4.4 Adaptive Algorithm for DecLQ

In this section, we propose an algorithm for the DecLQ problem, which is based on the certainty equivalence controller used in the literature for the centralized LQ problem. When the system model is unknown, the agent(s) need to learn the system model over time by interacting with the system through an adaptive algorithm. The learning usually happens by forming a belief over the system model and updating it over time. This belief will then be used to construct a controller, which is usually time-varying and adaptive due to the belief evolving by time. There are different ways that one can utilize the belief over the system model to construct the controller. Thompson Sampling (TS) [2, 4], Optimization in the Face of Uncertainty (OFU) [1, 45, 65], and Certainty Equivalence (CE) [114, 44] are the most well-known algorithms used in the literature. We note that since in CE, one only needs to have an estimation of the system model, it is sufficient to only keep track of the estimation as opposed to an entire belief over the system model. We will use CE to build our adaptive algorithm for the DecLQ problem. We refer to this algorithm

as the FedCE (Federated Certainty Equivalence) algorithm. Before describing FedCE algorithm, we highlight the main ideas used in its design and how they contribute to the overall performance of the algorithm.

Key Idea 1, Coordinated Exploration: It is evident that in our decentralized decoupled setting with partial state observation, the agents can not learn the system dynamics if both of them are exerting actions in all dimensions. The reason is that the actions are not shared with each other and the ambiguity in the control action of the other agent prevents them from learning the system. Instead, we devised coordinated exploration intervals in which, each agent only controls its own part of the state and stays silent for the other parts. During these intervals, we have a completely decoupled system in terms of dynamics and the agents can learn their corresponding system dynamics.

Key Idea 2, Federated Learning: In FedCE, agents learn the whole system model in a federated fashion. As mentioned, in the exploration intervals, the agents learn their corresponding model dynamics. However, in order to learn the optimal controller, the agents need the whole system dynamics. Therefore, they learn the other parts of the system dynamics through federated learning.

Key Idea 3, Balanced Exploration vs Exploitation: Due to the decentralized and decouples nature of our system model, the agents can only learn the system dynamics in the exploration intervals. Therefore, the longer the exploration intervals, the faster agents can learn the system dynamics. In addition, during the exploration intervals, the actions are not optimal or even near optimal. Therefore, the agents incur a linear regret over those intervals. Consequently, we need to set a balance between the exploration and exploitation interval durations.

4.4.1 FedCE Algorithm

FedCE is summarized in Alg. 5 and in the following, we will explain it in detail. FedCE algorithm consists of three different phases. A **Warm-up** period to form a good initial estimation

of the system model; **Certainty Equivalence (CE)** phase to exploit the model estimation and apply semi-optimal control actions; and **Exploration** phase to learn the system model. Note that CE and Exploration phases are distributed across separate time intervals.

Warm-up: In the Warm-up period, which starts from time $t = 0$ and ends at $t = T_w$ for some predetermined T_w, agents apply decoupled actions to enable decentralized learning of the system model (system identification). Decoupled actions are defined as

$$u_i(t) = \begin{bmatrix} 0 & \cdots & u_i^{(i)}(t) & \cdots & 0 \end{bmatrix}^T, \tag{4.5}$$

where $u_i^{(i)}(t) \sim \sigma_i^1$ and σ_i^1 is a predetermined probability distribution over $\mathbb{R}^{n_i}$.

During the warm-up period, each agent learns its corresponding system model using the least square estimate (LSE) method. In particular, agent i learns $\theta_i^* = [A_{ii}, B_{ii}]^T$ according to the sequential LSE method, in which the estimates on the system model, denoted by $\hat{\theta}_i(t)$, are updated sequentially for every new decoupled observation. We denote $\phi_i(t) = \begin{bmatrix} x_i(t) \\ u_i^{(i)}(t) \end{bmatrix}$ for $i \in \mathscr{N}$. The LSE update rule is as follows for $i \in \mathscr{N}$:

$$\hat{\theta}_i(t+1) = \hat{\theta}_i(t) + V_i(t)^{-1}\phi_i(t)(x_i(t+1)^T - \phi_i(t)^T\hat{\theta}_i(t))$$
$$V_i(t+1) = V_i(t) + \phi_i(t)\phi_i(t)^T$$

We define $\hat{\theta}_i$ to be the most up-to-date estimate on θ_i^* that agent i has at the current time (t). Note that although the agents have learned their corresponding system model parameters during the warm-up period, they have not communicated their estimates with each other yet. This communication will happen over the next phase. We define $\hat{\hat{\theta}} = [\hat{\hat{A}}, \hat{\hat{B}}]^T$ to be the most up-to-date estimate that all agents agree on. At the end of the warm-up period, we have $\hat{\hat{\theta}}$ to be the prior (initial) estimates of the agents over the model parameter, which we assume they all agree on.

Certainty Equivalence (CE): At the beginning of the CE phase, the agents will first compute the certainty equivalence controllers, $K(\hat{\hat{\theta}})$, using their latest model estimate that they all agree on which is denoted by $\hat{\hat{\theta}}$, using equations (4.2) and (4.3). We note that the CE controller is the optimal controller for a hypothetical system with system model equal to $\hat{\hat{\theta}}$. The control actions will be generated according to the computed CE controller. This phase is distributed into intervals of length T_{CE}^n, where n denotes the n^{th} CE interval. In the first D time slots of the CE phase, the agents also communicate and pass on their most up-to-date learned estimates on the model parameter, $\hat{\theta}_i$. At the end of the CE phase, all agents agree on the estimates $\hat{\theta}_i$ and they will update $\hat{\hat{\theta}}$ accordingly. We assume that $T_{CE} > D$. Each CE interval is followed by an interval of Exploration phase, which will be explained next.

Exploration: Exploration phase is similar to the warm-up phase but it is distributed into intervals of length T_{Exp}^n, where n denotes the n^{th} Exploration phase. Each exploration interval follows a CE interval and the agents apply the decoupling actions of the warm-up phase defined in (4.5) with $u_i^{(i)}(t) \sim \sigma_i^n$, and σ_i^n is a predetermined probability distribution over $\mathbb{R}^{n_i}$. We will specify the required conditions on σ_i^n in the next section. Over the exploration phase, the agents will learn their estimates $\hat{\theta}_i$ similar to the warm-up phase. Each exploration phase is followed by a CE phase.

The Exploration intervals should get sparser as time increases because the model estimates become better through time, and therefore, there is less need for exploration. In Section 4.5, we will describe the necessary conditions for the scheduling method and an example of such scheduling to have the desired regret behavior.

4.5 Convergence and Regret Analysis

In order to analyze the regret of FedCE algorithm, we first describe the conditions that we need to put on the exploration action distributions σ_i^n, for $i = \mathscr{N}$ and $n = 1, 2, \cdots$, and on the CE/Exploration interval scheduling; More specifically, the values of T_{CE}^n and T_{Exp}^n, for

Algorithm 5. FedCE

Inputs: $\hat{\hat{\theta}}$, σ_i^0, T_w, σ_i^n, T_{CE}^n, and T_{Exp}^n for $n = 1, 2, \cdots$.
for $t = 0, \cdots, T_w$ **do**
Let $u_i(t) = \begin{bmatrix} 0 & \cdots & u_i^{(i)}(t) & \cdots & 0 \end{bmatrix}^T$,
$u_i^{(i)}(t) \sim \sigma_i^0$, for $i \in \mathscr{N}$.
Agent i learns A_{ii}, B_{ii} using Eq. (4.6).
end for
for $n = 1, 2, \cdots$ **do**
Compute $K(\hat{\hat{\theta}})$ according to Eq. (4.2).
for $s = 1, \cdots, T_{CE}^n$ **do** $\implies$ *CE Phase*
$t = t + 1$.
Let

$$u_i(t) = \begin{bmatrix} K_{1i}(\theta^*) \\ \cdots \\ K_{Ni}(\theta^*) \end{bmatrix} x_i(t)$$

if $s \leq D$ **then** communicate and pass on $\hat{\theta}_i,\ i \in \mathscr{N}$ with the neighbors.
end if
end for
Agents update $\hat{\hat{\theta}}$ with their computed and received model estimates.
for $s = 1, \cdots, T_{Exp}^n$ **do** $\implies$ *Exploration Phase*
$t = t + 1$.
Let $u_i(t) = \begin{bmatrix} 0 & \cdots & u_i^{(i)}(t) & \cdots & 0 \end{bmatrix}^T$,
$u_i^{(i)}(t) \sim \sigma_i^n$, for $i \in \mathscr{N}$.
Agent i learns A_{ii}, B_{ii} using Eq. (4.6).
end for
end for

$n = 1, 2, \cdots$. We define $(\hat{u}_i^{(i)}(s))_{s=1,2,\cdots}$ to be agent i's control action sequence during Exploration intervals (s stands for the s_{th} exploration time). In particular, we have $\hat{u}_i^{(i)}(s) = u_i^{(i)}(t)$, where t corresponds to the s_{th} exploration time. We refer to $(\hat{u}_i^{(i)}(s))_{s=1,2,\cdots}$ as the exploration control action sequence.

The next two definitions are extracted from [87].

Definition 3 (Persistent Excitation)**.** *The exploration control action sequence $(\hat{u}_i^{(i)}(s))_{s=1,2,\cdots}$ is said to be persistently exciting if there exists a positive definite matrix U such that for large S, we have*

$$\frac{1}{S}\sum_{s=1}^{S} \hat{u}_i^{(i)}(s)\hat{u}_i^{(i)}(s)^T \geq U$$

The above definition states the condition when the exploration control actions are rich enough to facilitate the system model learning.

The next definition is about the stability of the system during the exploration intervals.

Definition 4 (Exploration Stability)**.** *The exploration control action sequence $(\hat{u}_i^{(i)}(s))_{s=1,2,\cdots}$ is said to satisfy the Exploration Stability condition if there exists a positive definite matrix V such that for large S, we have*

$$\frac{1}{S}\sum_{s=1}^{S} \hat{u}_i^{(i)}(s)\hat{u}_i^{(i)}(s)^T \leq V$$

Definition 5 (Rich and Stable Exploration)**.** *If Persistent Excitation and Exploration Stability conditions hold for the control sequence $(\hat{u}_i^{(i)}(s))_{s=1,2,\cdots}$ that is generated according to the distributions $\hat{u}_i^{(i)}(s) \sim \sigma_i^n$, when s^{th} exploration time belongs to the n^{th} exploration episode, then $(\sigma_i^n)_{n=1,2,\cdots}$ satisfies the Rich and Stable Exploration condition.*

Using the above definitions, we can have the following lemma on the system model learning rate of FedCE.

Lemma 3. *Let A^* and B^* be the true system model matrices, and $\hat{A}(s)$ and $\hat{B}(s)$ be least square mean estimates of A^* and B^* using s samples of the system (after s exploration times). Assume*

that the Rich and Stable Exploration condition holds for $(\sigma_i^n)_{n=1,2,\cdots}$ *for* $i = 1,2$. *Then with probability close to 1,* $||A^* - \hat{A}(s)||$ *and* $||B^* - \hat{B}(s)||$ *are bounded by* $\mathscr{O}(\sqrt{\frac{1}{s}})$ *for large enough s.*

Proof. According to [87], if Persistent Excitation and Exploration Stability conditions hold, then the estimation $\hat{\theta}_i(s)$ converges to the true system model θ_i^* almost surely, and we can write the following.

$$\hat{\theta}_i(s) = \theta_i^* + (\sum_{k=1}^{s} \phi_i(k)\phi_i(k)^T)^{-1} \sum_{k=1}^{s} \phi_i(k)w(k)^T \tag{4.7}$$

Persistent Excitation condition indicates that $sU \leq \sum_{k=1}^{s} \phi_i(k)\phi_i(k)^T \leq sV$ and consequently, $\lambda_{max}((\sum_{k=1}^{s} \phi_i(k)\phi_i(k)^T)^{-1}) \leq \frac{1}{s}l$ for some positive l, and $tr((\sum_{k=1}^{s} \phi_i(k)\phi_i(k)^T)) \leq sm$ for some positive m. By writing equation (4.7) for column c of $\hat{\theta}_i(s) - \theta_i^*$, denoted by $\hat{\theta}_i(s)[:,c] - \theta_i^*[:,c]$, and taking the L^2 norm, we have

$$\begin{aligned}
&||(\hat{\theta}_i(s)[:,c] - \theta_i^*[:,c])||_2^2 \\
&\quad \leq \frac{1}{s^2} l^2 (\sum_{k=1}^{s} \phi_i(k)^T w(k)[c])(\sum_{k=1}^{s} \phi_i(k)w(k)[c]) \\
&\quad \leq \frac{1}{s^2} l^2 (\sum_{k=1}^{s} \phi_i(k)^T \phi_i(k)w(k)[c]^2) \\
&\quad \leq \frac{1}{s^2} l^2 (\sum_{k=1}^{s} \phi_i(k)^T \phi_i(k))\gamma\sigma_w^2 \leq \frac{1}{s} l^2 m\gamma\sigma_w^2
\end{aligned}$$

The last two inequalities are valid with a probability close to 1, and this probability quickly approaches 1 as γ is increased. When γ is set to 4, the probability is nearly equal to 0.9999.

Therefore, $||A^* - \hat{A}(s)||$ and $||B^* - \hat{B}(s)||$ are bounded by $\mathscr{O}(\sqrt{\frac{1}{s}})$ for large enough s. □

In Lemma 3, we provided the learning rate with respect to the number of exploration times. The next theorem provides a bound on the learning rate with respect to the total time horizon, T.

Theorem 2 (Bounded Error)**.** *If Rich and Stable Exploration condition holds for* $(\sigma_i^n)_{n=1,2,\cdots}$ *and if the total number of exploration times is* $\mathcal{O}(\sqrt{T})$*, where* T *is the total time horizon, then* $||A^* - \hat{\hat{A}}(T)||$ *and* $||B^* - \hat{\hat{B}}(T)||$ *are bounded by* $\mathcal{O}(T^{-\frac{1}{4}})$.

We skip the proof of the above theorem since it is evident by using Lemma 3. The transition from $\hat{\theta}$ to $\hat{\hat{\theta}}$ does not affect the proof because $\hat{\hat{\theta}}$ is a delayed version of $\hat{\theta}$ with a constant time.

In the next theorem, we present the bound on the regret of the FedCE algorithm.

Theorem 3 (Regret Bound)**.** *Assume that Rich and Stable Exploration condition holds for* $(\sigma_i^n)_{n=1,2,\cdots}$ *for* $i \in \mathcal{N}$*. Also, assume that the total exploration time is* $\mathcal{O}(\sqrt{T})$ *for the total time horizon of* T*. Then the expected regret of the proposed FedCE algorithm satisfies*

$$\mathscr{R}(FedCE, T) = \mathcal{O}(\sqrt{T})$$

Proof. The regret of Algorithm 5 consists of two parts that we denote by $\mathscr{R}_{CE}$ and $\mathscr{R}_{Exp}$. In particular, we have

$$\mathscr{R}(\text{FedCE}, T) = \mathscr{R}_{CE}(\text{FedCE}, T) + \mathscr{R}_{Exp}(\text{FedCE}, T).$$

The first part, $\mathscr{R}_{CE}$, is related to CE intervals and the fact that we have an error in estimating the model and consequently, we have an error in constructing the optimal controller through certainty equivalence. The second part of the regret, $\mathscr{R}_{Exp}$, is related to Exploration phase due to not exerting the optimal action over those intervals. In the following, we will compute these two parts of the regret separately.

In order to compute $\mathscr{R}_{CE}$, we borrow some results from [114]. In particular, we use Theorem 2 of this reference, which gives an upper bound on the suboptimality of the certainty equivalence controller as a function of the model estimation error. In that theorem, they show that if the model estimation error is less than or equal to ε, then the average cost incurred through CE controller, denoted by J_{CE}, minus the optimal average cost, denoted by J^*, is bounded by a quadratic function of ε. We know from Theorem 2 that the model estimation error scales

as $\mathscr{O}(T^{\frac{-1}{4}})$. Therefore, $J_{CE} - J^*$ scales as $\mathscr{O}(T^{\frac{-1}{2}})$. From the definition of regret, we have $\mathscr{R}_{CE} = \mathscr{O}(T(J_{CE} - J^*))$. Therefore, we have $\mathscr{R}_{CE} = \mathscr{O}(T * T^{\frac{-1}{2}}) = \mathscr{O}(\sqrt{T})$.

It is easy to see that the second part of the regret, $\mathscr{R}_{Exp}$ scales linearly with the time of exploration. Note that this is the worst possible regret. The reason is that for every time step that we do not exert the optimal action, we pay a constant regret. Furthermore, assuming the state has deviated too much from the CE path (the path it would have taken if we were only applying CE controller), we can force it back to its path with at most the same time steps that have caused the deviation. Therefore, this part of the regret would be linear in the exploration time and we have $\mathscr{R}_{Exp} = \mathscr{O}(T_{Exp})$, where T_{Exp} is the total exploration time. Based on the assumption in the theorem, we have $T_{Exp} = \mathscr{O}(\sqrt{T})$. Consequently, we have $\mathscr{R}_{Exp} = \mathscr{O}(\sqrt{T})$.

Combining the above results for $\mathscr{R}_{CE}$ and $\mathscr{R}_{Exp}$, we have $\mathscr{R}(\text{FedCE}, T) = \mathscr{O}(\sqrt{T})$. □

4.5.1 Satisfying Rich and Stable Exploration

In this subsection, we provide an example of probability distributions $(\sigma_i^n)_{n=1,2,\cdots}$ for $i = 1, 2$, and CE/Exploration scheduling, i.e., T_{EX}^n and T_{Exp}^n for $n = 1, 2, \cdots$, that satisfies the Rich and Stable Exploration. According to Theorem 3, we also need to have the total exploration time to be $\mathscr{O}(\sqrt{T})$. Therefore, our example must also satisfy this condition.

Lemma 4. *If we have σ_i^n to be an element-wise Bernouli distribution over the set $\{a^n, -a^n\}$ with parameter $p^n = 0.5$, $a^n = \max\{\frac{a^1}{\sqrt{n}}, \delta\}$ for some positive a^1, and $T_{Exp} = kn$, then Rich and Stable Exploration condition holds.*

Proof. Consider the n^{th} exploration episode containing times $b+1, \cdots, b+T_{Exp}^n$. For large enough n, we have

$$\sum_{k=b+1}^{b+T_{Exp}^n} \hat{u}_i^{(i)}(k)\hat{u}_i^{(i)}(k)^T \approx T_{Exp}^n Cov(\hat{u}_i^{(i)}(b+1))$$

We also have

$$\delta^2 I_{n_i} \leq Cov(\hat{u}_i^{(i)}(b+1)) \leq (a^1)^2 I_{n_i}$$

Consequently, for large enough s, we can write

$$\delta^2 I_{n_i} \leq \frac{1}{s}\sum_{k=1}^{s} \hat{u}_i^{(i)}(k)\hat{u}_i^{(i)}(k)^T \leq (a^1)^2 I_{n_i}$$

Setting $U = \delta^2 I_{n_i}$ and $V = (a^1)^2 I_{n_i}$ will conclude the proof. □

Lemma 5. *If we set $T_{Exp}^n = kn$ and $T_{EX}^n = kn^3$, then the total exploration time is $\mathcal{O}(\sqrt{T})$.*

Proof. We denote the total time horizon with n CE/Exploration intervals with $T(n)$, and we have $T(n) = T_w + \sum_{l=1}^{n}(T_{CE}^n + T_{Exp}^n)$. We also denote the total exploration time in $T(n)$ by $T_{Exp}(n)$. According to the scheduling of $T_{CE}^n = kn^3$ and $T_{Exp}^n = kn$, we have $T(n) = \mathcal{O}(n^4)$, and $T_{Exp}(n) = \mathcal{O}(n^2)$. Therefore, we have $T_{Exp}(n) = \mathcal{O}(\sqrt{T(n)})$. We can drop n and write $T_{Exp} = \mathcal{O}(\sqrt{T})$, where T_{Exp} is the total exploration time in T total steps. □

4.6 Numerical Analysis

In this section, we illustrate the performance of our proposed FedCE algorithm through numerical simulations. We first consider the zero-target case, where the goal is to reach the reference $x^* = 0$. We then consider two more general cases: 1) nonzero target and 2) tracking trajectory generated by a known linear system of equations.

4.6.1 Zero-target control

In this section, we consider the problem of zero-target control, where the goal is to stabilize the system around the origin $x^* = 0$ in a gradual fashion. We use the following dynamics and cost matrices:

$$A = \begin{bmatrix} 0.95 & 0 \\ 0 & 0.95 \end{bmatrix}, B = I, Q = 0.05 * \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, R = I$$

We set the horizon to be $T = 40000$ and the length of the the warm-up phase $T_w = 20$. Each simulation setting is averaged over 100 independent runs, and the cumulative regret

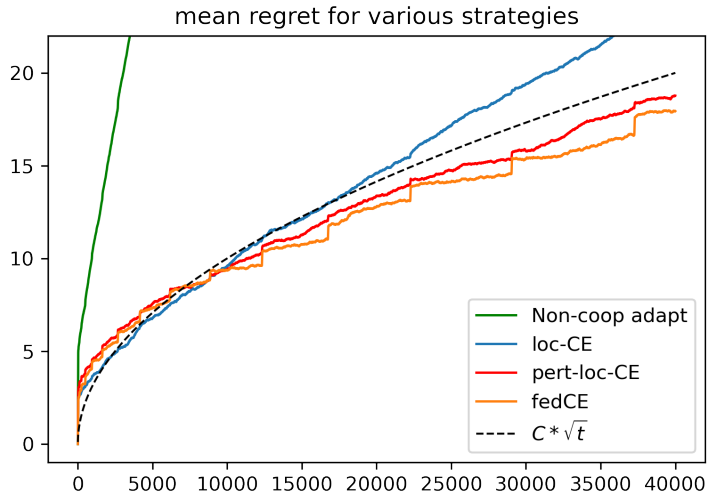
$\mathscr{R}(\text{FedCE}, T)$ and estimation error $||A - \hat{A}||$ are recorded. We set the initial state $x(0) = \mathbf{0}$ and the process noise variance $\sigma_w = 0.2$. We use the example of Section 4.5.1 for σ_i^n, T_{EX}^n, and T_{Exp}^n for FedCE algorithm (set $\delta = 0.01$ and consider different k and a^1 in our experiments). In our numerical analysis results, when we refer to FedCE, we are referring to this case. We further compare this version with cases when other alternative examples are utilized. We compare the performance of FedCE with the following alternative algorithms:

- **Non-cooperative Adaptive Algorithm (Non-Coop-Adapt)**: Similar to FedCE, except for the control policy applied in the CE intervals. Instead of applying CE controller, apply the sub-optimal control: $u_1(t) = \begin{bmatrix} K(\hat{\theta}^{11}) \\ 0 \end{bmatrix} x_1(t), u_2(t) = \begin{bmatrix} 0 \\ K(\hat{\theta}^{22}) \end{bmatrix} x_2(t)$. Where $K(\hat{\theta}^{11}), K(\hat{\theta}^{22})$ are computed from Eq.(4.2) with $(\hat{A}^{11}, \hat{B}^{11}, Q^{11}, R^{11})$, $(\hat{A}^{22}, \hat{B}^{22}, Q^{22}, R^{22})$ respectively.

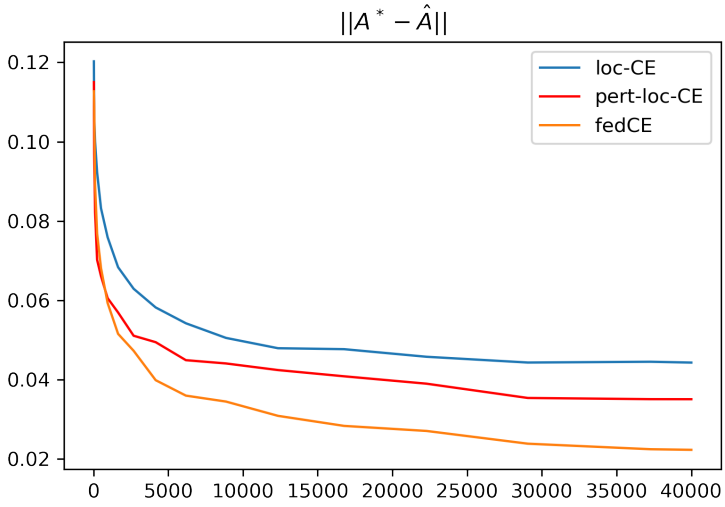
- **Local Certainty Equivalence (Loc-CE)**: Similar to FedCE but during Exploration intervals, set $u_1(t) = \begin{bmatrix} K^{11}(\hat{\theta}) \\ 0 \end{bmatrix} x_1(t), u_2(t) = \begin{bmatrix} 0 \\ K^{22}(\hat{\theta}) \end{bmatrix} x_2(t)$.

- **Perturbed Local Certainty Equivalence (Pert-Loc-CE)**: Similar to FedCE but during Exploration intervals, set $u_1(t) = \begin{bmatrix} K^{11}(\hat{\theta}) \\ 0 \end{bmatrix} x_1(t) + \begin{bmatrix} I \\ 0 \end{bmatrix} \eta_1(t), u_2(t) = \begin{bmatrix} 0 \\ K^{22}(\hat{\theta}) \end{bmatrix} x_2(t) + \begin{bmatrix} 0 \\ I \end{bmatrix} \eta_2(t)$, where $\eta_1(t), \eta_2(t) \sim \mathcal{N}(0, a^n)$ for the n^{th} exploration episode.

In Figure 4.1, we show the mean regret and model estimation error of FedCE and other alternative algorithms. For the analysis in this figure, we set $a^1 = 0.2$, $k = 2$ using a grid search. Figure 4.1(a) verifies the $\mathcal{O}(\sqrt{T})$ regret bound for FedCE as indicated by Theorem 3. We also show in this figure that the regret of **Non-Coop-Adapt** scales almost linearly, resulting from the fact that it treats the cost matrices as completely decoupled, incurring linear regret from the off-diagonal entries of Q, R. This result shows that sublinear regret is hopeless without



(a)



(b)

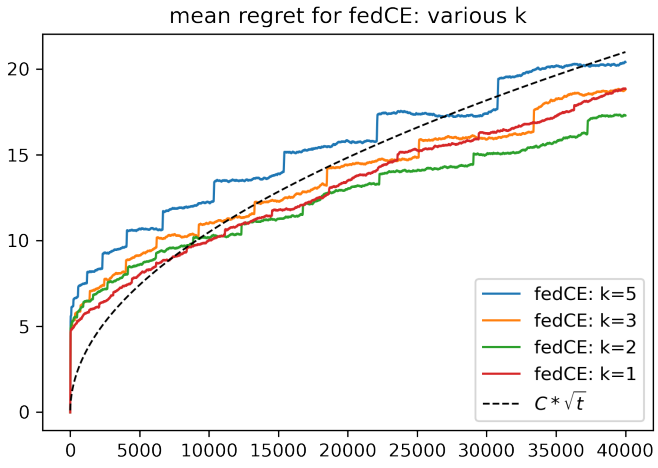
Figure 4.1. Performance of FedCE and other alternative algorithms. (a) Mean Regret, (b) Model Estimation Error.

communication between the two agents. Allowing communication significantly improves regret, as shown by **Loc-CE** and **Pert-Loc-CE** plots. However, the choice of exploration actions in **Loc-CE** provide insufficient excitation and therefore hinders the learning of system dynamics, as illustrated in 4.1(b). Although the added perturbations in **Pert-Loc-CE** improved the performance of the algorithm compared to **Loc-CE**, this exploration scheme performs much worse than FedCE. Since the exploration in **Loc-CE** and **Pert-Loc-CE** is based on CE controller, the actions are relatively "conservative", meaning the states are pulled back towards the lowest cost state (in this case $\mathbf{0}$) with little magnitude, causing smaller signal to noise ratios and consequently inaccurate system identification from least squares.

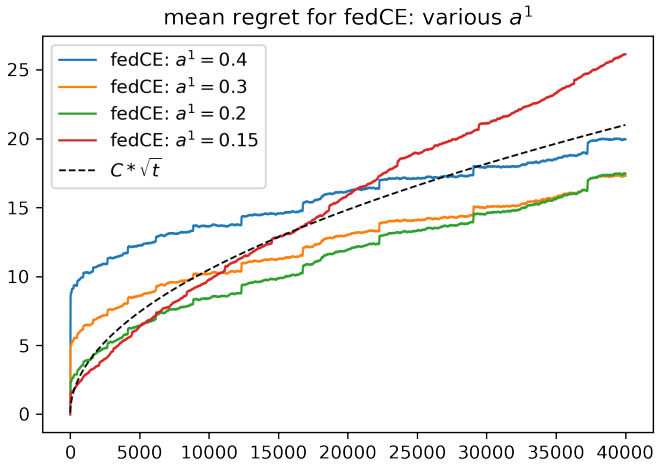
In Figure 4.2, we carry out an ablation study on Exploration/CE length scaling factor k and initial excitation a^1. In Figure 4.2(a), we study the effect of k on the regret. As k becomes smaller, exploration episodes are more frequent but less lengthy, creating a trade-off between the exploration regret and system learning. In Figure 4.2(b), FedCE is initialized with different a^1. Similarly, there appears to be a trade-off between system model learning (better with larger a^1) and the regret per exploration period (more with larger a^1).

In Figure 4.3, we apply FedCE to the same system as Figure 4.1, with the following differences. $x(0) = [0, 10]'$, and $Q = 0.05 * \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$. For this system the lowest cost state is when $x_1 = x_2$. We study the effect of R, i.e. cost matrix for actions, on the speed by which x_1 and x_2 converge to each other. As R takes larger magnitudes, larger actions are more penalized, therefore x_1 and x_2 converge more slowly since it is preferred to take several smaller steps instead of a giant leap.

In Figure 4.4, we apply FedCE to an unstable system. with $A = \begin{bmatrix} 1.05 & 0 \\ 0 & 1.05 \end{bmatrix}$. Due to the instability of the system, each exploration phase incurs more regret since random actions can bring the system further from the lowest cost state. Despite this fact, FedCE is still able to achieve a sublinear regret bounded by $\mathcal{O}(\sqrt{T})$, and the rate of learning is of the order of $\mathcal{O}(T^{-\frac{1}{4}})$



(a)



(b)

Figure 4.2. Ablation studies of FedCE under different (**a**) scaling factor k, and (**b**) initial excitation a^1.

that matches the theoretical results from Theorem 2.

4.6.2 Beyond zero-target Decentralized LQ Problems

In sections 4.4 and 4.5, the problem of Decentralized LQ control with zero target is introduced and analyzed. Most existing literature in LQ control considers this zero target case, where the goal of the controller is to gradually bring the system to the zero state and to maintain that state over the infinite horizon. However, in real-world applications, the target is often

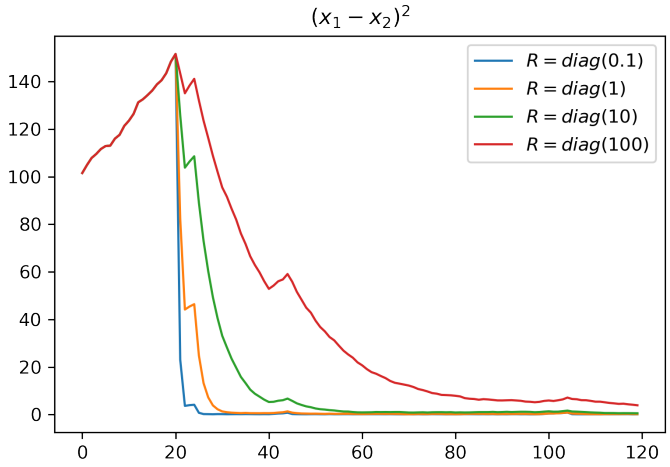


Figure 4.3. Convergence of x_1 and x_2 to each other for various R.

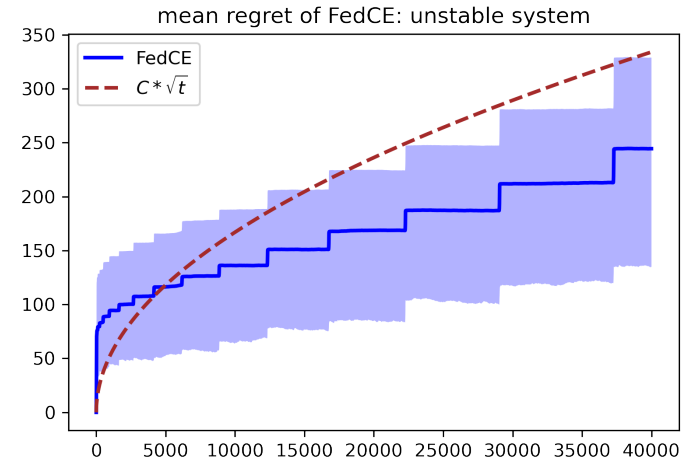
nonzero, or even time-variant. For instance, in building thermal control problems, one might want to maintain the temperature at a certain level; for drones to robotic navigation systems, we might want to follow a certain trajectory or maintain some kind of formations.

Driven by these potential application scenarios, we propose three extensions to the aforementioned Decentralized LQ problem with zero target. In the first case, the agents of the system aim to track fixed nonzero point references. In the second case, the agents aim to follow respective trajectories generated by a known linear dynamic model. In the last case, the agents aim to track fixed nonzero point references, while trying to maintain certain distance from each other. The algorithms for these problems are variants of FedCE, where they differ slightly in the Certainty Equivalence (CE) control phase.

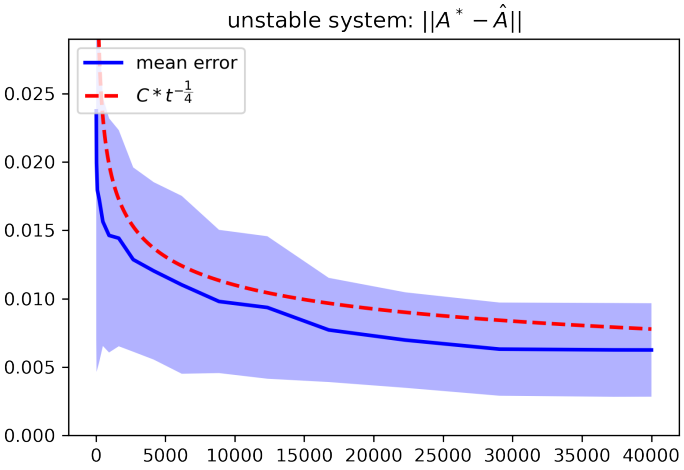
Waypoint Target Control

In this setting, each agent tracks its respective non-zero reference points. Formally, given state $x(t)$ and joint control $u(t)$, as defined in section 4.3, the agents receive a quadratic step cost of

$$c(t) = (x(t) - x^*(t))^T Q(x(t) - x^*(t)) + u(t)^T R u(t) \tag{4.8}$$



(a)



(b)

Figure 4.4. Performance of FedCE for an unstable system.

where $x^*(t) := \begin{bmatrix} x_1^*(t) \\ x_2^*(t) \end{bmatrix}$ is the reference point for the system and has the same dimension as $x(t)$, $x^*(t) \in \mathbb{R}^{n_1+n_2}$. Since we are considering a fixed reference point, we can drop the dependence on t, so $x^*(t) = x^*$.

The goal is to find control policies $\pi(t) = (\pi_1(t), \pi_2(t))$, where $\pi_i(t) : \mathbb{R}^{n_i} \to \mathbb{R}^d$ and $u_i(t) = \pi_i(t)[x_i(t)]$, that minimizes the infinite-horizon average expected cost

$$J^{\pi}(\theta) = \limsup_{T\to\infty} \frac{1}{T}\mathbb{E}^{\pi}[\sum_{t=1}^{T} c(t)]$$

In the following lemma, we present the optimal controllers when the system model $\theta^* = [A^*, B^*]^T$ is known. This controller will be the basis of the Certainty Equivalence (CE) control phase in FedCE.

In the Decentralized LQ problem with nonzero reference point, if the system model $\theta^* = [A^*, B^*]^T$ is known, the optimal controllers are given by [9]

$$u(t) = K(\theta^*)x(t) + \bar{K}(\theta^*)x^*$$

where

$$K(\theta^*) = -(R + B^T P(\theta^*)B)^{-1}B^T P(\theta^*)A,$$

and

$$\bar{K}(\theta^*) = -(R + B^T P(\theta^*)B)^{-1}B^T (I - (A + BK))^{-1}Q,$$

and $P(\theta^*)$ is the solution to the Algebraic Riccati Equation (4.3).

Therefore if we denote $K(\theta^*) = \begin{bmatrix} K_{11}(\theta^*) & K_{12}(\theta^*) \\ K_{21}(\theta^*) & K_{22}(\theta^*) \end{bmatrix}$,

and $\bar{K}(\theta^*) = \begin{bmatrix} \bar{K}_{11}(\theta^*) & \bar{K}_{12}(\theta^*) \\ \bar{K}_{21}(\theta^*) & \bar{K}_{22}(\theta^*) \end{bmatrix}$, we have the optimal control for the two agents as:

$$u_1(t) = \begin{bmatrix} K_{11}(\theta^*) \\ K_{21}(\theta^*) \end{bmatrix} x_1(t) + \begin{bmatrix} \bar{K}_{11}(\theta^*) \\ \bar{K}_{21}(\theta^*) \end{bmatrix} x_1^*,$$

$$u_2(t) = \begin{bmatrix} K_{12}(\theta^*) \\ K_{22}(\theta^*) \end{bmatrix} x_2(t) + \begin{bmatrix} \bar{K}_{12}(\theta^*) \\ \bar{K}_{22}(\theta^*) \end{bmatrix} x_2^*,$$

The above can be proved from Corollary 3.

Simulations: We designated 3 waypoints for each agent. Each agent should aim to reach each waypoint before a certain time limit while minimizing cumulative costs. Specifically, the

time limits for the 3 waypoints are $[500, 1000, 1500]$, and corresponding waypoint coordinates are: $x_1^* = [(1,2),(2,1),(2.5,2.5)]$ and $x_2^* = [(1,0.5),(2,1.5),(2.5,2)]$. We assume that the agents are moving at different (fixed) heights, simulating drone movements. See Figure 4.5), where we have plotted the trajectory of two agents with FedCE against reference trajectories generated if $[A^*, B^*]$ are known in advance.

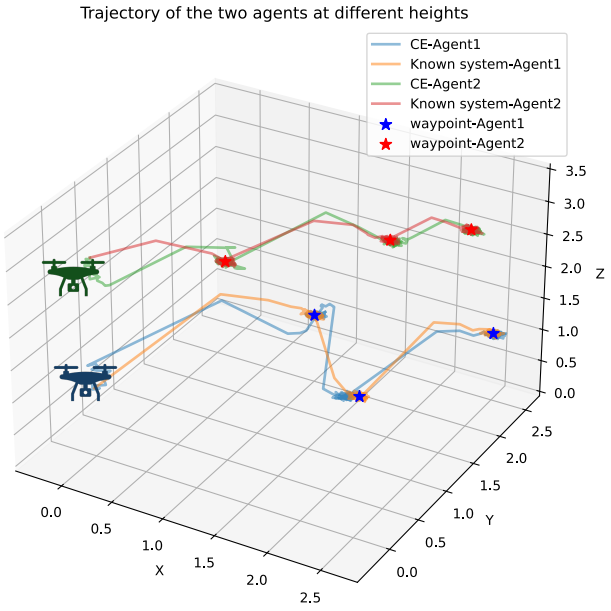


Figure 4.5. Trajectory of two agents in the case of nonzero reference.

In Figure 4.5, we plotted the trajectory of two agents with FedCE against reference trajectories generated if $[A^*, B^*]$ are known in advance. In Figure 4.6, we plotted the cumulative regret and observed that it is bounded by $\mathcal{O}(\sqrt{T})$.

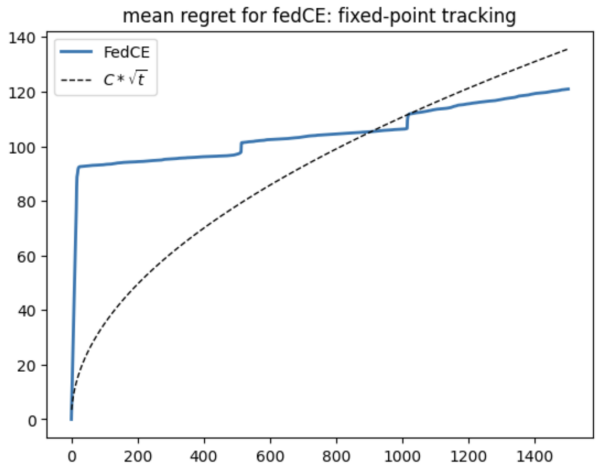


Figure 4.6. Cumulative regret in the case of nonzero reference.

Route Tracking Control

In this setting, each agent tracks a trajectory generated by a known linear system, with state equations defined as:

$$z(t+1) = Fz(t), x^*(t) = Hz(t)$$

Same as in previous section, given state $x(t)$ and joint control $u(t)$, as defined in section 4.3, the agents receive a quadratic step cost of

$$c(t) = (x(t) - x^*(t))^T Q(x(t) - x^*(t)) + u(t)^T Ru(t) \tag{4.9}$$

where $x^*(t) := \begin{bmatrix} x_1^*(t) \\ x_2^*(t) \end{bmatrix}$ is the reference point for the system and has the same dimension as $x(t)$, $x^*(t) \in \mathbb{R}^{n_1+n_2}$. In the following lemma, we present the optimal controllers when the system model $\theta^* = [A^*, B^*]^T$ is known. This controller will be the basis of the Certainty Equivalence (CE) control phase in FedCE.

Lemma 6. *In the Decentralized LQ tracking problem with trajectory generated from a known linear system, if the system model $\theta^* = [A^*, B^*]^T$ is known, one can first construct an auxiliary*

system defined as [9]:

$$\hat{A} = \begin{bmatrix} A & 0 \\ 0 & F \end{bmatrix}, \hat{B} = \begin{bmatrix} B \\ 0 \end{bmatrix}, \hat{Q} = \begin{bmatrix} Q & -QH \\ -H^T Q & H^T QH \end{bmatrix}$$

then the optimal control is given by:

$$u(t) = K_1 x(t) + K_2 z(t)$$

where

$$K_1 = -R^{-1} B^T P$$

$$K_2 = -R^{-1} B^T P_{12}$$

where P is from $\hat{P} = \begin{bmatrix} P & P_{12} \\ P_{21} & P_{22} \end{bmatrix}$, where $\hat{P}$ is calculated from the Discrete Algebraic Ricatti Equations with the auxiliary system:

$$P = \hat{Q} + \hat{A}^T P \hat{A} - \hat{A}^T P \hat{B} (R + \hat{B}^T P \hat{B})^{-1} \hat{B}^T P \hat{A}.$$

Therefore, if we denote $K_1(\theta^*) = [K_{1,\mathbf{1}}(\theta^*), K_{1,\mathbf{2}}(\theta^*)]$,
and $K_2(\theta^*) = [K_{2,\mathbf{1}}(\theta^*), K_{2,\mathbf{2}}(\theta^*)]$, we have the optimal control for the two agents as:

$$u_1(t) = K_{1,\mathbf{1}}(\theta^*) x_1(t) + K_{2,\mathbf{1}}(\theta^*) z_1(t),$$

$$u_2(t) = K_{1,\mathbf{2}}(\theta^*) x_2(t) + K_{2,\mathbf{2}}(\theta^*) z_2(t),$$

Certainty Equivalence (CE): In the CE phase, the agents will compute the certainty equivalence controllers, $K_1(\hat{\theta}), K_2(\hat{\theta})$, using their latest model estimate, $\hat{\theta}$, using equations (6) and (6).

Simulations: For the simulation, each agent aims to track a trajectory generated from a

known linear system. We use $F = \begin{bmatrix} F_1 & 0 \\ 0 & F_2 \end{bmatrix}$, and $H = \mathbf{I}$, so the two trajectories are decoupled. We let $F_1 = F_2 = \begin{bmatrix} 0.98 & 0.1 \\ -0.1 & 0.75 \end{bmatrix}$ and $x_1(0) = (0.1, 0.1)$, $x_2(0) = (0.2, 0.2)$. In Figure 4.7 and 4.8, we plotted the trajectory of two agents with FedCE against trajectories generated if $[A^*, B^*]$ are known in advance.

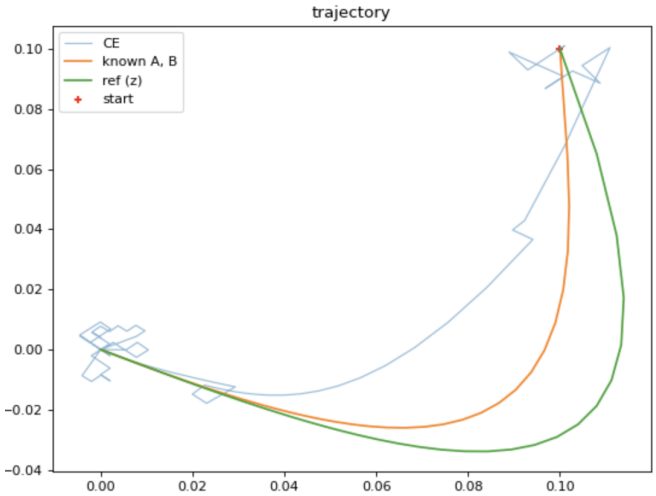


Figure 4.7. Trajectory of agent 1 in the case of tracking a trajectory generated from a known linear system.

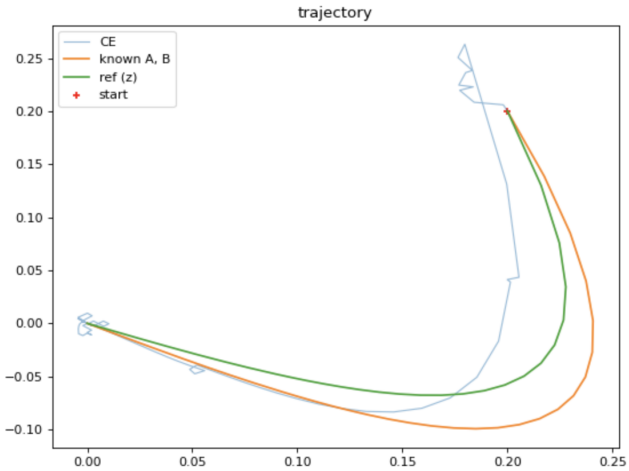


Figure 4.8. Trajectory of agent 2 in the case of tracking a trajectory generated from a known linear system.

4.7 Conclusions and Future Work

In this chapter, we studied the challenging decentralized linear quadratic control problem with unknown transition dynamics, where agents observe states partially and can control partial inputs over an additive channel. We proposed FedCE, an algorithm that balances system identification with least squares (exploration) and certainty equivalence control (exploitation). Theoretically, we showed that given Rich and Stable Explorations, FedCE achieves a regret that scales at a rate of $\mathcal{O}(\sqrt{T})$ for a time horizon of T (Theorem 3), and gave an example such that the Rich and Stable Explorations can be achieved. Numerical simulations completed our analysis by confirming the theoretical results, and provided ablation studies for various parameters in the algorithm.

One natural extension to the DecLQ problem is coupled dynamics, where A, B matrices are dense instead of block-diagonal. In this case agents have to perform system identification with partial observations, which might involve *Ho-Kalman* type algorithms [59, 91]. Moreover, DecLQ with more complex global controls instead of pure addition of local inputs is also an interesting direction for future work, and new algorithms can be developed that exploits the structure of global controls. Finally, settings where observed states are corrupted, such as by an adversarial attacker, or settings that require reliability and safety guarantees, are also crucial for future investigations.

4.8 Acknowledgements

Chapter 4, in part, is a reprint of the material published in the 59th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 2023, pp. 1-8: "Federated Certainty Equivalence Control for Linear Gaussian Systems with Unknown Decoupled Dynamics and Quadratic Common Cost" (Xinghan Wang, Nasimeh Heydaribeni, Farinaz Koushanfar and Tara Javidi). The dissertation author was the primary investigator and author of this paper. This research is partially supported by NSF TILOS AI Institute,

AutoCOMBOT MURI Grant, and ONR Award N00014-22-1-2363.

Chapter 5

Conclusion and Future Directions

This thesis presents three case studies of managing and modeling uncertainties in machine learning. In the supervised learning setting, we focus on *epistemic uncertainty* (model uncertainty) by employing a Bayesian learning framework and studying models' convergence behavior under a peer-to-peer decentralized learning setting. Our proposed method combines *Variational Bayes* and *Bayesian merge rule*, allowing local models to eventually learn the true model parameters under potentially *non-IID* local data. In the reinforcement learning (RL) setting, we consider both *aleatoric* and *epistemic* uncertainties. In RL problems, *aleatoric* uncertainty can potentially come from stochastic transitions in environments, while *epistemic* uncertainty usually comes from insufficient exploration of the environment. For the two RL case studies, we take *model-based* approaches: we first derive optimal policies under known stochastic transition models, then design efficient exploration strategies to guarantee sufficient exploration of the environment. The keys to designing exploration strategies are sufficient exploration phases and adequate estimation strategy of unknown model parameters.

There are many potential future directions for each of the three cases considered in this thesis. For the decentralized learning problem, common challenges include: 1) adversarial attacks [23], 2) node dropouts [61], 3) local distribution shifts or sometimes called *drifts* [134]. Given the rapid development of new machine learning models, it is also interesting to consider Bayesian decentralized learning for *Large Language Models* (LLM) [167], *Diffusion Models*

[162], or *Foundation Models* [26]. For the Contextual Shortest Path problem, learning under more complex context distributions can be an extension of the presented work. For the decentralized LQ problem, one direction is to go from LQR to MDP and devise strategies for a decentralized MDP problem. One can also study other model-based approaches, such as *Thompson Sampling*. In addition, the problem setup can be extended to coupled dynamics and beyond additional joint controls.

Appendix A

Appendix to Chapter 2

A.1 Bernstein-Von Mises Theorem under model misspecification

Theorem 4. *Let $q^{(t)}(\theta \mid \mathscr{D}) \in \mathscr{P}(\Theta)$ be the posterior distribution after observing $\mathscr{D}_{1:t} = \{(\mathbf{X}_s, \mathbf{Y}_s)\}_{1\le s\le t} = \{(X_s^{(i)}, Y_s^{(i)}) \in (\mathscr{X}, \mathscr{Y})\}_{i\in[n], 1\le s\le t}$, where $\mathscr{D}_{1:t}$ is drawn i.i.d. from $\mathsf{P}_{XY} = \mathsf{P}_X\mathsf{P}_{Y|X}$. Let $p_\theta \in \mathscr{P}(\Theta)$ be the prior distribution over Θ. The Bernstein-Von Mises Theorem asserts that, for any measurable set $A \subset \Theta$, we have,*

$$\sup_A \mid q^{(t)}(A) - N_{\hat{\theta}_t, (tV_{\theta^*})^{-1}}(A) \mid \xrightarrow{\mathsf{P}_{XY}} 0$$

where

$$\theta^* \in \operatorname*{argmin}_{\theta\in\Theta} \mathbb{E}_{\mathsf{P}_X}[D_{KL}(\mathsf{P}_{Y|X}(\cdot \mid X) || f(\cdot \mid X, \theta))]$$

and V_{θ^*} is the negative Hessian matrix of the above expected KL divergence, and $\hat{\theta}_t$ is some suitable estimators, typically taken as the maximum likelihood estimator that satisfies the sequence $\sqrt{t}(\hat{\theta}_t - \theta^*)$ is asymptotically normal with zero mean. Here, it is not required that there exists a $\theta^* \in \Theta$ such that $f(\cdot \mid X, \theta^*) = \mathsf{P}_{Y|X}(\cdot \mid X)$ almost surely under P_X. In other words, Bernstein-Von Mises Theorem under model misspecification guarantees convergence of the posterior regardless of realizability.

A.2 Consensus Step on Gaussian distributions

Let $(\mu_i^{(t)}, \Sigma_i^{(t)})$ denote the mean and the covariance matrix of $\mathbf{b}_i^{(t)}$ at agent i at time t. Using equation (2.4), we have

$$\begin{aligned}
&\sum_{j=1}^{N} W_{ij} \ln G(\theta, \mu_j^{(t)}, \Sigma_j^{(t)}) \\
&= -\frac{1}{2} \sum_{j=1}^{N} W_{ij} \left((\theta - \mu_j^{(t)})^{\mathrm{T}} {\Sigma_j^{(t)}}^{-1} (\theta - \mu_j^{(t)}) \right) \\
&\quad - \frac{1}{2} \sum_{i=1}^{N} W_{ij} \ln(2\pi)^k |\Sigma_j^{(t)}| \\
&= -\frac{1}{2} \left(\theta^T \sum_{j=1}^{N} W_{ij} {\Sigma_j^{(t)}}^{-1} \theta + \sum_{j=1}^{N} {\mu_j^{(t)}}^T W_{ij} {\Sigma_j^{(t)}}^{-1} \mu_j^{(t)} \right) \\
&\quad + \frac{1}{2} \left(\sum_{j=1}^{N} {\mu_j^{(t)}}^T W_{ij} {\Sigma_j^{(t)}}^{-1} \theta + \theta^T \sum_{j=1}^{N} W_{ij} {\Sigma_j^{(t)}}^{-1} \mu_i^{(t)} \right) \\
&\quad - \frac{1}{2} \sum_{j=1}^{N} W_{ij} \ln(2\pi)^k |\Sigma_j^{(t)}|.
\end{aligned}$$

Noting that $(\theta - \mathbf{A})^T \Sigma^{-1} (\theta - \mathbf{A}) = \theta^T \Sigma^{-1} \theta - \mathbf{A}^T \Sigma^{-1} \theta - \theta^T \Sigma^{-1} \mathbf{A} + \mathbf{A}^T \Sigma^{-1} \mathbf{A}$. By completing the squares we obtain $\mathbf{q}_i^{(t)}$ is Gaussian distribution with the last term becoming part of the normalization constant after exponentiating. and we have

$$\widetilde{\Sigma}_i^{(t)^{-1}} = \sum_{j=1}^{N} W_{ij} {\Sigma_j^{(t)}}^{-1},$$

and

$$\begin{aligned}
\widetilde{\Sigma}_i^{(t)^{-1}} \widetilde{\mu}_i^{(t)} &= \sum_{j=1}^{N} W_{ij} {\Sigma^{(t)}}_j^{-1} \mu_j^{(t)} \\
\implies \widetilde{\mu}_i^{(t)} &= \widetilde{\Sigma}_i^{(t)} \sum_{j=1}^{N} W_{ij} {\Sigma_j^{(t)}}^{-1} \mu_j^{(t)}.
\end{aligned}$$

A.3 Proof of Theorem 1

Before proving the theorem we first present a lemma on the stochastic matrix W.

Lemma 7 ([141]). *For an irreducible and aperiodic stochastic matrix W, the stationary distribution $\mathbf{v} = [v_1, v_2, \ldots, v_N]$ is unique and has strictly positive components and satisfies*

$v_i = \sum_{j=1}^{n} v_j W_{ji}$. *Furthermore, for any* $i \in [N]$ *the weight matrix satisfies*

$$\sum_{k=1}^{n}\sum_{j=1}^{N}\left|[W^k]_{ij} - v_j\right| \leq \frac{4\log N}{1-\lambda_{max}(W)},$$

where $\lambda_{max}(W) = \max_{i\in[N-1]} \lambda_i(W)$*, and* $\lambda_i(W)$ *denotes eigenvalue of* W *counted with algebraic multiplicity and* $\lambda_0(W) = 1$.

The proof of Theorem 1 is based the proof provided in [124, 141, 92]. For the ease of exposition, let $b_i^{(0)}(\theta) = \frac{1}{|\Theta|}$ for all $\theta \in \Theta$. We begin with the following recursion for each node $i \in [N]$ and for any $\theta \in \Theta \setminus \mathscr{B}_r(\theta^*)$,

$$\frac{1}{T}\log\frac{b_i^{(T)}(\theta^*)}{b_i^{(T)}(\theta)} = \frac{1}{T}\sum_{j=1}^{N}\sum_{t=0}^{T}[W^t]_{ij} z_j^{(T-t)}(\theta^*,\theta),$$

where

$$z_j^{(t)}(\theta^*,\theta) = \log\frac{f^{(j)}\left(\mathbf{Y}_t^{(j)} \mid \mathbf{X}_t^{(j)}, \theta^*\right)}{f^{(j)}\left(\mathbf{Y}_t^{(j)} \mid \mathbf{X}_t^{(j)}, \theta\right)}.$$

From the above recursion, we have

$$\begin{aligned}
\frac{1}{T}\log\frac{b_i^{(T)}(\theta^*)}{b_i^{(T)}(\theta)} &= \frac{1}{T}\sum_{j=1}^{N} v_j\left(\sum_{t=0}^{T} z_j^{(T-t)}(\theta^*,\theta)\right) \\
&\quad + \frac{1}{T}\sum_{j=1}^{N}\left(\sum_{t=0}^{T}\left([W^t]_{ij} - v_j\right) z_j^{(T-t)}(\theta^*,\theta)\right) \\
&\geq \frac{1}{T}\sum_{j=1}^{N} v_j\left(\sum_{t=0}^{T} z_j^{(t)}(\theta^*,\theta)\right) \\
&\quad - \frac{1}{T}\sum_{j=1}^{N}\sum_{t=0}^{T}\left|[W^t]_{ij} - v_j\right|\left|z_j^{(t)}(\theta^*,\theta)\right| \\
&\overset{(a)}{\geq} \frac{1}{T}\sum_{j=1}^{N} v_j\left(\sum_{t=0}^{T} z_j^{(t)}(\theta^*,\theta)\right) - \frac{4C\log N}{T(1-\lambda_{\max}(W))}
\end{aligned}$$

where (a) follows from Lemma 7 and the boundedness assumption of log-likelihood ratios. Now fix $T \geq \frac{8C\log N}{\varepsilon(1-\lambda_{\max}(W))}$, since $b_i^{(T)}(\theta^*) \leq 1$ (since $b_i^{(T)}(\cdot)$ is a pmf over Θ) we have

$$-\frac{1}{T}\log b_i^{(T)}(\theta) \geq -\frac{\varepsilon}{2} + \frac{1}{T}\sum_{j=1}^{N} v_j\left(\sum_{t=0}^{T} z_j^{(t)}(\theta^*,\theta)\right).$$

Furthermore, we have

$$\mathsf{P}\left(-\frac{1}{T}\log b_i^{(T)}(\theta) \le \sum_{j=1}^{N} v_j I_j(\theta^*,\theta) - \varepsilon\right)$$
$$\le \mathsf{P}\left(\frac{1}{T}\sum_{j=1}^{N} v_j \sum_{t=0}^{T} z_j^{(t)}(\theta^*,\theta) \le \sum_{j=1}^{N} v_j I_j(\theta^*,\theta) - \frac{\varepsilon}{2}\right).$$

where recall that

$$\begin{aligned} I_j(\theta^*,\theta) &= \mathbb{E}[z_j(\theta^*,\theta)] \\ &= \mathbb{E}_{\mathsf{P}_X^{(j)}}\Big[D_{\mathrm{KL}}\left(\mathsf{P}_{Y|X}(\cdot|\mathbf{X})||f^{(j)}(\cdot \mid \theta,\mathbf{X})\right) \\ &\quad - D_{\mathrm{KL}}\left(\mathsf{P}_{Y|X}(\cdot|\mathbf{X})||f^{(j)}(\cdot \mid \theta^*,\mathbf{X})\right)\Big]. \end{aligned}$$

Now for any $j \in [N]$ note that

$$\sum_{j=1}^{N} v_j \sum_{t=0}^{T} z_j^{(t)}(\theta^*,\theta) - T\sum_{j=1}^{N} v_j I_j(\theta^*,\theta)$$
$$= \sum_{t=0}^{T}\left(\sum_{j=1}^{N} v_j z_j^{(t)}(\theta^*,\theta) - \sum_{j=1}^{N} v_j \mathbb{E}[z_j^{(t)}(\theta^*,\theta)]\right).$$

For any $\theta \notin \mathscr{B}_r(\theta^*)$, applying McDiarmid's inequality for all $\varepsilon > 0$ and for all $T \ge 1$ we have

$$\mathsf{P}\left(\sum_{t=0}^{T}\left(\sum_{j=1}^{N} v_j z_j^{(t)}(\theta^*,\theta) - \sum_{j=1}^{N} v_j \mathbb{E}[z_j^{(t)}(\theta^*,\theta)]\right) \le -\frac{\varepsilon T}{2}\right)$$
$$\le e^{-\frac{\varepsilon^2 T}{2C}}.$$

Hence, for all $\theta \notin \mathscr{B}_r(\theta^*)$, for $T \ge \frac{8C\log N}{\varepsilon(1-\lambda_{\max}(W))}$ we have

$$\mathsf{P}\left(\frac{-1}{T}\log b_i^{(T)}(\theta) \le \sum_{j=1}^{N} v_j I_j(\theta^*,\theta) - \varepsilon\right) \le e^{-\frac{\varepsilon^2 T}{4C}},$$

which implies

$$\mathsf{P}\left(b_i^{(T)}(\theta) \ge e^{-T(\sum_{j=1}^{N} v_j I_j(\theta^*,\theta)-\varepsilon)}\right) \le e^{-\frac{\varepsilon^2 T}{4C}}.$$

Using this we obtain a bound on the worst-case error over all θ and across the entire network as follows

$$\mathsf{P}\left(\max_{i\in[N]}\max_{\theta\in\Theta\setminus\mathscr{B}_r(\theta^*)} b_i^{(T)}(\theta) \ge e^{-T(K(\Theta)-\varepsilon)}\right) \le N|\Theta|e^{-\frac{\varepsilon^2 T}{4C}},$$

where $K(\Theta) := \min_{\theta\in\Theta\setminus\mathscr{B}_r(\theta^*)} \sum_{j=1}^{N} v_j I_j(\theta^*,\theta)$. From Lemma 7 we have that $K(\Theta) > 0$. Then, with probability at least $1-\delta$ we have

$$\max_{i\in[N]}\max_{\theta\in\Theta\setminus\mathscr{B}_r(\theta^*)} b_i^{(T)}(\theta) < e^{-T(K(\Theta)-\varepsilon)},$$

when the number of samples satisfies

$$T \geq \frac{4C \log \frac{N|\Theta|}{\delta}}{\varepsilon^2}.$$

However, recall we require $T \geq \frac{8C \log N}{\varepsilon(1-\lambda_{\max}(W))}$, a reasonable lower bound is therefore

$$T \geq \frac{8C \log \frac{N|\Theta|}{\delta}}{\varepsilon^2(1-\lambda_{\max}(W))}.$$

A.4 Proof of Corollary 1

$$\begin{aligned}
&\frac{1}{N}\sum_{i=1}^{N} |L_i(\theta^*) - L_i(\hat{\theta}_i^{(T)})| \\
&\leq \frac{B}{N}\sum_{i=1}^{N} \mathbb{E}_{\mathsf{P}_X}\left[\int_{\mathscr{Y}} \left| f^{(i)}(\hat{y} \mid x, \theta^*) - f^{(i)}(\hat{y} \mid x, \hat{\theta}_i^{(T)}) \right| dy\right] \\
&\leq \frac{B}{2N}\sum_{i=1}^{N} \mathbb{E}_{\mathsf{P}_X}\left[\sqrt{D_{KL}(f^{(i)}(\hat{y} \mid x, \theta^*)||f^{(i)}(\hat{y} \mid x, \hat{\theta}_i^{(T)}))}\right] \\
&\leq \frac{B}{2}\sqrt{\frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\mathsf{P}_X}\left[D_{KL}(f^{(i)}(\hat{y} \mid x, \theta^*)||f^{(i)}(\hat{y} \mid x, \hat{\theta}_i^{(T)}))\right]} \\
&\leq \frac{B\sqrt{r}}{2}
\end{aligned}$$

where the third to last inequality follows from Pinsker's Inequality, the second to last inequality follows from Jensen's Inequality, and the last line follows from Theorem 1 and Assumption 3.

Appendix B

Appendix to Chapter 3

Proof of Theorem 1 The regret of RL-CSP algorithm consists of two parts: $R^{explore}(T)$, the regret incurred from exploration phases, and $R^{exploit}(T)$, the regret from exploitation phases before the value functions have not converged which results in sub-optimal rank ordering of states.

Bounding $R^{explore}(T)$: Note that the number of exploration phases is bounded by $\mathcal{O}(g\log(t))$. For each exploration phase, a state can be at most M transitions away from the source. Therefore, the total regret from exploration is bounded by $\mathcal{O}(Ng\log(T))$.

Bounding $R^{exploit}(T)$: Let $V_{\hat{\mathbf{P}}}(i)$ be the optimal value function of state i under empirical context distribution $\hat{\mathbf{P}}_i^a = \hat{P}(X|i,a)$. It is shown in [67] that $|V^*(i) - V_{\hat{\mathbf{P}}}(i)| \leq \alpha \max_{i,a} \sigma(\mathbf{P}_i^a, \hat{\mathbf{P}}_i^a)$, where $\alpha = R\sum_{i=1}^{N}\sum_u \sum_a \frac{1}{c_U(i,X,u)+c_A(i,a)}$ and σ is the total variation metric. For exploitation episode t, define $L(t)$ as the event that the policy under $V_{\hat{\mathbf{P}}}(i)$ is different from the policy under $V^*(i)$. Notice that if $|V^*(i) - V_{\hat{\mathbf{P}}}(i)| \leq \frac{\Delta}{2}, \forall i \in \mathscr{V}$, then the optimal policy under $\hat{\mathbf{P}}_i^a$ and $\mathbf{P}_i^a$ are the same. Therefore, $\{\max_{i,a} \sigma(\mathbf{P}_i^a, \hat{\mathbf{P}}_i^a(t)) \leq \Delta/2\alpha\} \subseteq L^c(t)$. Thus we have:

$$Pr(L(t)) \leq \sum_a \sum_{i=1}^{N} Pr(\sigma(\mathbf{P}_i^a, \hat{\mathbf{P}}_i^a(t)) > \Delta/2\alpha)$$

Now within the summation on the right, we have:

$$
\begin{aligned}
&Pr(\sigma(\mathbf{P}_i^a, \hat{\mathbf{P}}_i^a(t)) > \Delta/2\alpha) \\
&= Pr(\frac{1}{2}\sum_{X \in \mathscr{X}_i} |\hat{P}_t(X|i,a) - P(X|i,a)| > \Delta/2\alpha) \\
&\leq \sum_{X \in \mathscr{X}_i} Pr(|\hat{P}_t(X|i,a) - P(X|i,a)| > \frac{\Delta}{B\alpha})
\end{aligned}
$$

Let $n_i(t)$ be the number of times state i is visited upon time t, then by applying a Chernoff Bound we obtain:

$$
Pr(\sigma(\mathbf{P}_i^a, \hat{\mathbf{P}}_i^a(t)) > \Delta/2\alpha) \leq 2Bexp(-2n_i(t)(\frac{\Delta}{B\alpha})^2)
$$

Since by time t, there are at least $g\log t$ exploration episodes, where for each exploration episode the least visited state is reached, we have:

$$
n_i(t) \geq g/Mlog(t) \geq \alpha^2 B^2/(2\Delta^2)log(t), \forall i \in \mathscr{V}
$$

Thus we have:

$$
\begin{aligned}
Pr(\sigma(\mathbf{P}_i^a, \hat{\mathbf{P}}_i^a(t)) > \Delta/2\alpha) &\leq 2Bexp(-2n_i(t)(\frac{\Delta}{B\alpha})^2) \\
&\leq \frac{2B}{t}
\end{aligned}
$$

$$
Pr(L(t)) \leq \sum_a \sum_{k=1}^{M} \frac{2B}{t} \leq \frac{2BM|\mathscr{A}|}{t}
$$

Thus the regret from exploitation can be bounded as:

$$
R^{exploit}(T) \leq R\sum_t Pr(L(t)) \leq 2RMB|\mathscr{A}|(log(T))
$$

Now, recall $\alpha = \mathscr{O}(M)$, so $R^{explore}(T) = \mathscr{O}(M^4B^2\log(T)/\Delta^2)$ which is a larger bound than $2RMB|\mathscr{A}|(\log(T))$. Thus, $R(T) = \mathscr{O}(Mg\log(T))$.

Appendix C

Appendix to Chapter 4

Proof of Lemma 2. When θ^* is known, one can look at DecLQ problem from a centralized agent's point of view. Meaning, assume we have a centralized agent that observes both $x_1(t)$ and $x_2(t)$ and generates the control action $u(t)$. Therefore, we know that the optimal controller will be $u(t) = K(\theta^*)x(t)$. Therefore, we can write

$$u(t) = \sum_{i=1}^{N} \begin{bmatrix} K_{1i}(\theta^*) \\ \dots \\ K_{Ni}(\theta^*) \end{bmatrix} x_i(t),$$

by setting $u_i(t) = \begin{bmatrix} K_{1i}(\theta^*) \\ \dots \\ K_{Ni}(\theta^*) \end{bmatrix} x_i(t)$, we can achieve the above optimal control. Therefore, in a decentralized case, if θ^* is known, all of the agents can compute the matrix $K(\theta^*)$ and generate their corresponding control actions that will collectively achieve the optimal cost of the centralized controller, $J^*(\theta^*)$ (which is the best one can achieve in a decentralized setting). □

Bibliography

[1] Yasin Abbasi-Yadkori and Csaba Szepesvári. Regret bounds for the adaptive control of linear quadratic systems. In Sham M. Kakade and Ulrike von Luxburg, editors, *Proceedings of the 24th Annual Conference on Learning Theory*, volume 19 of *Proceedings of Machine Learning Research*, pages 1–26, Budapest, Hungary, 09–11 Jun 2011. PMLR.

[2] Yasin Abbasi-Yadkori and Csaba Szepesvári. Bayesian optimal control of smoothly parameterized systems. In *UAI*, pages 1–11, 2015.

[3] Naoki Abe and Philip M Long. Associative reinforcement learning using linear probabilistic concepts. In *ICML*, pages 3–11. Citeseer, 1999.

[4] Marc Abeille and Alessandro Lazaric. Improved regret bounds for thompson sampling in linear quadratic control problems. In *International Conference on Machine Learning*, 2018.

[5] Marco Aggravi, Giuseppe Sirignano, Paolo Robuffo Giordano, and Claudio Pacchierotti. Decentralized control of a heterogeneous human–robot team for exploration and patrolling. *IEEE Transactions on Automation Science and Engineering*, 19(4):3109–3125, 2021.

[6] Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *International conference on machine learning*, pages 127–135. PMLR, 2013.

[7] Maruan Al-Shedivat, Jennifer Gillenwater, Eric Xing, and Afshin Rostamizadeh. Federated learning via posterior averaging: A new perspective and practical algorithms, 2021.

[8] Christopher Amato, George Konidaris, Gabriel Cruz, Christopher A Maynor, Jonathan P How, and Leslie P Kaelbling. Planning for decentralized control of multiple robots under uncertainty. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 1241–1248. IEEE, 2015.

[9] B.D.O. Anderson and J.B. Moore. *Optimal Control: Linear Quadratic Methods*. Dover Books on Engineering. Dover Publications, 2007.

[10] Seyed Mohammad Asghari, Mukul Gagrani, and Ashutosh Nayyar. Regret analysis for learning in a multi-agent linear-quadratic control problem. In *2020 American Control Conference (ACC)*, pages 3926–3931. IEEE, 2020.

[11] Seyed Mohammad Asghari, Yi Ouyang, and Ashutosh Nayyar. Regret bounds for decentralized learning in cooperative multi-agent dynamical systems. In *Conference on Uncertainty in Artificial Intelligence*, pages 121–130. PMLR, 2020.

[12] Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Mike Rabbat. Stochastic gradient push for distributed deep learning. In *International Conference on Machine Learning*, pages 344–353. PMLR, 2019.

[13] P Auer. Finite-time analysis of the multiarmed bandit problem, 2002.

[14] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

[15] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

[16] Peter Auer, Thomas Jaksch, and Ronald Ortner. Near-optimal regret bounds for reinforcement learning. *Advances in neural information processing systems*, 21, 2008.

[17] Peter Auer and Ronald Ortner. Logarithmic online regret bounds for undiscounted reinforcement learning. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, page 49–56, Cambridge, MA, USA, 2006. MIT Press.

[18] Baruch Awerbuch and Robert Kleinberg. Online linear optimization and adaptive routing. *Journal of Computer and System Sciences*, 74(1):97–114, 2008.

[19] Y. Bar-Shalom and E. Tse. Dual effect, certainty equivalence, and separation in stochastic control. *IEEE Transactions on Automatic Control*, 19(5):494–500, 1974.

[20] Françoise Simone Beaufays, Mingqing Chen, Rajiv Mathews, and Tom Ouyang. Federated learning of out-of-vocabulary words. *arXiv: Computation and Language*, 2019.

[21] Richard Bellman. On a routing problem. *Quarterly of applied mathematics*, 16(1):87–90, 1958.

[22] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International conference on machine learning*, pages 634–643. PMLR, 2019.

[23] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International conference on machine learning*, pages 634–643. PMLR, 2019.

[24] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[25] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 1613–1622. JMLR.org, 2015.

[26] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[27] L. Bottou, F. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

[28] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6):2508–2530, 2006.

[29] Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

[30] Tamara Broderick, Nicholas Boyd, Andre Wibisono, Ashia C Wilson, and Michael I Jordan. Streaming variational bayes. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1727–1735. Curran Associates, Inc., 2013.

[31] Robert R Bush and Frederick Mosteller. A stochastic model with applications to learning. *The Annals of Mathematical Statistics*, pages 559–585, 1953.

[32] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

[33] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings, 2019.

[34] Trevor Campbell and Jonathan P How. Decentralized variational bayesian inference. *CoRR*, 2014.

[35] Nicolo Cesa-Bianchi and Gábor Lugosi. Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422, 2012.

[36] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 24, 2011.

[37] Pratik Chaudhari, Carlo Baldassi, Riccardo Zecchina, Stefano Soatto, and Ameet Talwalkar. Parle: parallelizing stochastic gradient descent. *CoRR*, abs/1707.00424, 2017.

[38] Han-Fu Chen and Lei Guo. *Identification and Stochastic Adaptive Control*. Birkhauser Boston Inc., USA, 1991.

[39] Mingzhe Chen, Nir Shlezinger, H Vincent Poor, Yonina C Eldar, and Shuguang Cui. Communication-efficient federated learning. *Proceedings of the National Academy of Sciences*, 118(17):e2024789118, 2021.

[40] Alon Cohen, Tomer Koren, and Yishay Mansour. Learning linear-quadratic regulators efficiently with only $\sqrt{T}$ regret. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1300–1309. PMLR, 09–15 Jun 2019.

[41] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1(1):269–271, December 1959.

[42] Peter Dorato, Vito Cerone, and Chaouki Abdallah. *Linear-Quadratic Control: An Introduction*. Simon amp; Schuster, Inc., USA, 1994.

[43] John C. Duchi, Alekh Agarwal, and Martin J. Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic Control*, 57:592–606, 2012.

[44] Mohamad Kazem Shirani Faradonbeh, Ambuj Tewari, and George Michailidis. Input perturbations for adaptive control and learning. *Automatica*, 117:108950, 2020.

[45] Mohamad Kazem Shirani Faradonbeh, Ambuj Tewari, and George Michailidis. Optimism-based adaptive regulation of linear-quadratic systems. *IEEE Transactions on Automatic Control*, 66(4):1802–1808, 2020.

[46] Shirani Faradonbeh, Ambuj Tewari, and George Michailidis. Input perturbations for adaptive control and learning. *Automatica*, 117:108950, 2020.

[47] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. *Advances in neural information processing systems*, 23, 2010.

[48] Lester R Ford Jr. Network flow theory. Technical report, Rand Corp Santa Monica Ca, 1956.

[49] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods, 2018.

[50] Takako Fujiwara-Greve. *Non-cooperative game theory*. Springer, 2015.

[51] Mukul Gagrani and Ashutosh Nayyar. Thompson sampling for some decentralized control problems. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 1053–1058. IEEE, 2018.

[52] Mukul Gagrani, Sagar Sudhakara, Aditya Mahajan, Ashutosh Nayyar, and Yi Ouyang. A modified thompson sampling-based learning algorithm for unknown linear systems. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 6658–6665, 2022.

[53] Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Transactions on Networking*, 20(5):1466–1478, 2012.

[54] Yarin Gal. Uncertainty in Deep Learning. *University of Cambridge*, 2016.

[55] Anousheh Gholami, Nariman Torkzaban, and John S Baras. Trusted decentralized federated learning. In *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–6. IEEE, 2022.

[56] Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

[57] Ben Hambly, Renyuan Xu, and Huining Yang. Recent advances in reinforcement learning in finance. *Mathematical Finance*, 33(3):437–503, 2023.

[58] Peng Hang, Chen Lv, Yang Xing, Chao Huang, and Zhongxu Hu. Human-like decision making for autonomous driving: A noncooperative game theoretic approach. *IEEE Transactions on Intelligent Transportation Systems*, 22(4):2076–2087, 2020.

[59] . L. Ho and Rudolf E. Kálmán. Editorial: Effective construction of linear state-variable models from input/output functions. 1965.

[60] Junya Honda and Akimichi Takemura. Optimality of thompson sampling for gaussian bandits depends on priors. In *Artificial Intelligence and Statistics*, pages 375–383. PMLR, 2014.

[61] Samuel Horvath, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos Venieris, and Nicholas Lane. Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout. *Advances in Neural Information Processing Systems*, 34:12876–12889, 2021.

[62] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip B. Gibbons. The non-iid data quagmire of decentralized machine learning. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[63] Bin Hu, Kaiqing Zhang, Na Li, Mehran Mesbahi, Maryam Fazel, and Tamer Başar. Toward a theoretical foundation of policy optimization for learning control policies. *Annual Review of Control, Robotics, and Autonomous Systems*, 6(1):123–158, 2023.

[64] Osband Ian, Van Roy Benjamin, and Russo Daniel. (more) efficient reinforcement learning via posterior sampling. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3003–3011, Red Hook, NY, USA, 2013. Curran Associates Inc.

[65] Morteza Ibrahimi, Adel Javanmard, and Benjamin Roy. Efficient reinforcement learning for high dimensional linear quadratic systems. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[66] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE symposium on security and privacy (SP)*, pages 19–35. IEEE, 2018.

[67] Tara Javidi and Demosthenis Teneketzis. Sensitivity analysis of an optimal routing policy in an ad hoc wireless network. *IEEE Transactions on Automatic Control*, 49(8):1303–1316, 2004.

[68] Zhanhong Jiang, Aditya Balu, Chinmay Hegde, and Soumik Sarkar. Collaborative deep learning in fixed topology networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5904–5914. Curran Associates, Inc., 2017.

[69] Zhanhong Jiang, Aditya Balu, Chinmay Hegde, and Soumik Sarkar. Collaborative deep learning in fixed topology networks. *Advances in Neural Information Processing Systems*, 30, 2017.

[70] Hao Jin, Yang Peng, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Federated reinforcement learning with environment heterogeneity. In *International Conference on Artificial Intelligence and Statistics*, pages 18–37. PMLR, 2022.

[71] Peter H. Jin, Qiaochu Yuan, Forrest N. Iandola, and Kurt Keutzer. How to scale distributed deep learning? *CoRR*, abs/1611.04581, 2016.

[72] Kwang-Sung Jun, Aniruddha Bhargava, Robert Nowak, and Rebecca Willett. Scalable generalized linear bandits: Online computation and hashing. *Advances in Neural Information Processing Systems*, 30, 2017.

[73] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning, 2021.

[74] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

[75] Shivam Kalra, Junfeng Wen, Jesse C Cresswell, Maksims Volkovs, and Hamid R Tizhoosh. Decentralized federated learning through proxy model sharing. *Nature communications*, 14(1):2899, 2023.

[76] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49:209–232, 2002.

[77] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 482–491, 2003.

[78] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[79] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[80] Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2575–2583. Curran Associates, Inc., 2015.

[81] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):4909–4926, 2021.

[82] B.J.K. Kleijn and A.W. van der Vaart. The bernstein-von-mises theorem under misspecification. *Electron. J. Statist.*, 6:354–381, 2012.

[83] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[84] Jakub Konecný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *CoRR*, abs/1610.02527, 2016.

[85] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated Learning: Strategies for Improving Communication Efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[86] Panqanamala Ramana Kumar. A survey of some results in stochastic adaptive control. *SIAM Journal on Control and Optimization*, 23(3):329–380, 1985.

[87] PR Kumar and Pravin Varaiya. Stochastic systems: estimation, identification and adaptive control, 1986.

[88] Tze Leung Lai. Adaptive treatment allocation and the multi-armed bandit problem. *The annals of statistics*, pages 1091–1114, 1987.

[89] Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

[90] Sahin Lale, Kamyar Azizzadenesheli, Babak Hassibi, and Anima Anandkumar. Explore more and improve regret in linear quadratic regulators. *ArXiv*, abs/2007.12291, 2020.

[91] Sahin Lale, Kamyar Azizzadenesheli, Babak Hassibi, and Anima Anandkumar. Adaptive control and regret minimization in linear quadratic gaussian (lqg) setting. In *2021 American Control Conference (ACC)*, pages 2517–2522, 2021.

[92] A. Lalitha, T. Javidi, and A. D. Sarwate. Social Learning and Distributed Hypothesis Testing. *IEEE Transactions on Information Theory*, 64(9):6161–6179, Sept 2018.

[93] Anusha Lalitha, Osman Cihan Kilinc, Tara Javidi, and Farinaz Koushanfar. Peer-to-peer federated learning on graphs, 2019.

[94] Anusha Lalitha, Anand Sarwate, and Tara Javidi. Social learning and distributed hypothesis testing. In *2014 IEEE International Symposium on Information Theory*, pages 551–555, 2014.

[95] Anusha Lalitha, Shubhanshu Shekhar, Tara Javidi, and Farinaz Koushanfar. Fully decentralized federated learning. In *Third NeurIPS workshop on Bayesian Deep Learning*, 2018.

[96] Peter Chal Landgren. *Distributed multi-agent multi-armed bandits*. PhD thesis, Princeton University, 2019.

[97] John Langford and Tong Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. *Advances in neural information processing systems*, 20(1):96–1, 2007.

[98] Martin Lauer and Martin A Riedmiller. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *Proceedings of the seventeenth international conference on machine learning*, pages 535–542, 2000.

[99] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[100] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. Technical report, 2021.

[101] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3347–3366, 2021.

[102] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

[103] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5330–5340. Curran Associates, Inc., 2017.

[104] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. Asynchronous decentralized parallel stochastic gradient descent. In *International Conference on Machine Learning*, pages 3043–3052. PMLR, 2018.

[105] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

[106] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE communications surveys & tutorials*, 22(3):2031–2063, 2020.

[107] Tao Lin, Sebastian U. Stich, and Martin Jaggi. Don't use large mini-batches, use local SGD. *CoRR*, abs/1808.07217, 2018.

[108] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

[109] Michael L Littman. Value-function reinforcement learning in markov games. *Cognitive systems research*, 2(1):55–66, 2001.

[110] K. Liu and Q. Zhao. Adaptive shortest-path routing under unknown and stochastically varying link states. In *2012 10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, pages 232–237, 2012.

[111] Owen Lockwood and Mei Si. A review of uncertainty for deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 18, pages 155–162, 2022.

[112] Guodong Long, Yue Tan, Jing Jiang, and Chengqi Zhang. Federated learning for open banking. In *Federated learning: privacy and incentive*, pages 240–254. Springer, 2020.

[113] Dhruv Malik, Ashwin Pananjady, Kush Bhatia, Koulik Khamaru, Peter Bartlett, and Martin Wainwright. Derivative-free methods for policy optimization: Guarantees for linear quadratic systems. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 2916–2925. PMLR, 16–18 Apr 2019.

[114] Horia Mania, Stephen Tu, and Benjamin Recht. Certainty equivalence is efficient for linear quadratic control. *Advances in Neural Information Processing Systems*, 32, 2019.

[115] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 2017.

[116] Volodymyr Mnih. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[117] Aditya Modi, Nan Jiang, Satinder Singh, and Ambuj Tewari. Markov decision processes with continuous side information. volume 83 of *Proceedings of Machine Learning Research*, pages 597–618. PMLR, 07–09 Apr 2018.

[118] Thomas M Moerland, Joost Broekens, Aske Plaat, Catholijn M Jonker, et al. Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 16(1):1–118, 2023.

[119] Nader Motee, Ali Jadbabaie, and Bassam Bamieh. On decentralized optimal control and information structures. In *2008 American Control Conference*, pages 4985–4990. IEEE, 2008.

[120] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640, 2021.

[121] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[122] Angelia Nedić and Alex Olshevsky. Stochastic gradient-push for strongly convex functions on time-varying directed graphs. *IEEE Transactions on Automatic Control*, 61(12):3936–3947, 2016.

[123] Angelia Nedić, Alex Olshevsky, and Michael G. Rabbat. Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106(5):953–976, 2018.

[124] Angelia Nedić, Alex Olshevsky, and César A. Uribe. Fast convergence rates for distributed non-bayesian learning. *IEEE Transactions on Automatic Control*, 62(11):5538–5553, 2017.

[125] Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

[126] Dinh C Nguyen, Quoc-Viet Pham, Pubudu N Pathirana, Ming Ding, Aruna Seneviratne, Zihuai Lin, Octavia Dobre, and Won-Joo Hwang. Federated learning for smart healthcare: A survey. *ACM Computing Surveys (Csur)*, 55(3):1–37, 2022.

[127] Saurabh Bagchi Nourian and Arash. Federated learning: How private is it really?, Aug 2023.

[128] Frans A Oliehoek, Christopher Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

[129] Yi Ouyang, Mukul Gagrani, Ashutosh Nayyar, and Rahul Jain. Learning unknown markov decision processes: A thompson sampling approach. *Advances in neural information processing systems*, 30, 2017.

[130] Christodoulos Pappas, Dimitris Chatzopoulos, Spyros Lalis, and Manolis Vavalis. Ipls: A framework for decentralized federated learning. In *2021 IFIP Networking Conference (IFIP Networking)*, pages 1–6. IEEE, 2021.

[131] My Phan, Yasin Abbasi-Yadkori, and Justin Domke. Thompson sampling with approximate inference, 2020.

[132] Athanasios S Polydoros and Lazaros Nalpantidis. Survey of model-based reinforcement learning: Applications on robotics. *Journal of Intelligent & Robotic Systems*, 86(2):153–173, 2017.

[133] Yuben Qu, Haipeng Dai, Yan Zhuang, Jiafa Chen, Chao Dong, Fan Wu, and Song Guo. Decentralized federated learning for uav networks: Architecture, challenges, and opportunities. *IEEE Network*, 35(6):156–162, 2021.

[134] Amirhossein Reisizadeh, Farzan Farnia, Ramtin Pedarsani, and Ali Jadbabaie. Robust federated learning: The case of affine distribution shifts. *Advances in Neural Information Processing Systems*, 33:21554–21565, 2020.

[135] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2021–2031. PMLR, 26–28 Aug 2020.

[136] Herbert Robbins. Some aspects of the sequential design of experiments. 1952.

[137] Aviv Rosenberg and Yishay Mansour. Online stochastic shortest path with bandit feedback and unknown transition function. In *Advances in Neural Information Processing Systems*, pages 2212–2221, 2019.

[138] Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

[139] Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

[140] Osama Shahid, Seyedamin Pouriyeh, Reza M. Parizi, Quan Z. Sheng, Gautam Srivastava, and Liang Zhao. Communication efficiency in federated learning: Achievements and challenges, 2021.

[141] S. Shahrampour, A. Rakhlin, and A. Jadbabaie. Distributed Detection: Finite-Time Analysis and Impact of Network Topology. *IEEE Transactions on Automatic Control*, 61(11):3256–3268, Nov 2016.

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

[143] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[144] Satinder Singh, Michael J Kearns, and Yishay Mansour. Nash convergence of gradient dynamics in general-sum games. In *UAI*, pages 541–548, 2000.

[145] Virginia Smith, Simone Forte, Chenxin Ma, Martin Takáč, Michael I. Jordan, and Martin Jaggi. CoCoA: A general framework for communication-efficient distributed optimization. *Journal of Machine Learning Research*, 18(230):1–49, 2018.

[146] Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in neural information processing systems*, 8, 1995.

[147] Richard S Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

[148] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. d^2: Decentralized training over decentralized data. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4848–4856, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[149] Zhenheng Tang, Shaohuai Shi, Bo Li, and Xiaowen Chu. Gossipfl: A decentralized federated learning framework with sparsified and adaptive communication. *IEEE Transactions on Parallel and Distributed Systems*, 34(3):909–922, 2022.

[150] Cem Tekin and Mihaela Van Der Schaar. Distributed online learning via cooperative contextual bandits. *IEEE transactions on signal processing*, 63(14):3700–3714, 2015.

[151] Ambuj Tewari and Susan A Murphy. From ads to interventions: Contextual bandits in mobile health. *Mobile health: sensors, analytic methods, and applications*, pages 495–517, 2017.

[152] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

[153] Eugene Vinitsky, Nathan Lichtlé, Kanaad Parvate, and Alexandre Bayen. Optimizing mixed autonomy traffic flow with decentralized autonomous vehicles and multi-agent reinforcement learning. *ACM Transactions on Cyber-Physical Systems*, 7(2):1–22, 2023.

[154] Xiaofei Wang, Yiwen Han, Chenyang Wang, Qiyang Zhao, Xu Chen, and Min Chen. In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning. *Ieee Network*, 33(5):156–165, 2019.

[155] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

[156] E. Wei and A. Ozdaglar. Distributed alternating direction method of multipliers. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5445–5450, Dec 2012.

[157] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE transactions on information forensics and security*, 15:3454–3469, 2020.

[158] Xiaofan Wu, Florian Dörfler, and Mihailo R Jovanović. Input-output analysis and decentralized optimal control of inter-area oscillations in power systems. *IEEE Transactions on Power Systems*, 31(3):2434–2444, 2015.

[159] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. 2017.

[160] Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Systems Control Letters*, 53(1):65–78, 2004.

[161] Jie Xu, Benjamin S Glicksberg, Chang Su, Peter Walker, Jiang Bian, and Fei Wang. Federated learning for healthcare informatics. *Journal of healthcare informatics research*, 5:1–19, 2021.

[162] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4):1–39, 2023.

[163] Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

[164] Serdar Yüksel and Tamer Basar. Stochastic networked control systems. *AMC*, 10:12, 2013.

[165] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pages 321–384, 2021.

[166] Jiaqi Zhao, Hui Zhu, Fengwei Wang, Rongxing Lu, Zhe Liu, and Hui Li. Pvd-fl: A privacy-preserving and verifiable decentralized federated learning framework. *IEEE Transactions on Information Forensics and Security*, 17:2059–2073, 2022.

[167] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[168] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[169] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey, 2021.

[170] Ruihao Zhu and Eytan Modiano. Learning to route efficiently with end-to-end feedback: The value of networked structure, 2018.

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

[172] Alexander Zimin and Gergely Neu. Online learning in episodic markovian decision processes by relative entropy policy search. In *Advances in neural information processing systems*, pages 1583–1591, 2013.