# Data Efficient Reinforcement Learning

by

Zhi Xu

B.S. in Electrical Engineering, University of Illinois at Urbana-Champaign, May 2015

S.M. in Electrical Engineering and Computer Science, Massachusetts Institute of Technology, June 2017

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2021



Signature of Author: _____

Zhi Xu
Department of Electrical Engineering and Computer Science
May 18, 2021

Certified by: _____

Devavrat Shah
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by: _____

Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

**Data Efficient Reinforcement Learning**

by

Zhi Xu

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electrical Engineering and Computer Science

**Abstract**

Reinforcement learning (RL) has recently emerged as a generic yet powerful solution for learning complex decision-making policies, providing the key foundational underpinnings of recent successes in various domains, such as game playing and robotics. However, many state-of-the-art algorithms are data-hungry and computationally expensive, requiring large amounts of data to succeed. While this is possible for certain scenarios, in applications arising in social sciences and healthcare for example, where available data is sparse, this naturally can be costly or infeasible. With the surging interest in applying RL to broader domains, it is imperative to develop an informed view about the usage of data involved in its algorithmic design.

This thesis hence focuses on studying the data efficiency of RL, through a structural perspective. Advancement along this direction naturally requires us to understand when and why algorithms are successful to begin with; and building upon such understanding, further improve the data efficiency of RL. To this end, this thesis begins by taking inspiration from the empirical successes. We consider the popular use of simulation-based Monte Carlo Tree Search (MCTS) in RL, as exemplified by the remarkable achievement of AlphaGo Zero, and probe the data efficiency of incorporating such a key ingredient. Specifically, we investigate the correct form to utilize such a tree structure for estimating values and characterize the corresponding data complexity. These results further enable us to analyze the data complexity of a RL algorithm that combines MCTS with supervised learning as done in AlphaGo Zero.

Having developed a better understanding, as a next step, we improve the algorithmic designs of simulation-based data-efficient RL algorithms that have access to a generative model. We provide such improvements for both bounded and unbounded spaces. Our first contribution is a structural framework through a novel lens of low-rank representation of the $Q$-function. The proposed data-efficient RL algorithm exploits the low-rank structure to perform pseudo-exploration by querying/simulating only a selected subset of state-action pairs, via a new matrix estimation technique. Remarkably, this leads to a significant (exponential) improvement in data complexity. Moving to our endeavor with unbounded spaces, one must first address the unique conceptual challenges incurred by the unbounded domains. Inspired by classical queueing systems, we propose an appropriate notion of stability for quantifying "goodness" of policies. Subsequently, by leveraging the stability structure of the underlying systems, we design efficient, adaptive algorithms with a modified, efficient

Monte Carlo oracle that guarantee the desired stability with a favorable data complexity that is polynomial with respect to the parameters of interest.

Altogether, through new analytical tools and structural frameworks, this thesis contributes to the design and analysis of data-efficient RL algorithms.

---

Thesis Supervisor: Devavrat Shah
Title: Professor of Electrical Engineering and Computer Science

# Acknowledgments

Six years ago, I was very fortunate to have the opportunity of choosing among several graduate schools. Deep in my heart, I knew that I could never say no to this three-letter university. Six years later, I am extremely glad that I made the right decision. Indeed, my MIT journey is wonderful, and I would like to take this opportunity to thank many people who have made it such an enjoyable and memorable experience.

First and foremost, I am extremely grateful to have Devavrat as my advisor. He is an amazing advisor who truly cares about his students and always put his students on top of the priority queue. Devavrat once mentioned that after graduation, my status would change from a student to a friend. But even before that, I have always felt that he is more like a friend to me. Of course, a wise and insightful friend. I absolutely enjoy drinking coffee together and just talking about random things. Whenever I have questions, I would simply text or call him, and he is always available for discussions. Needless to say, academically, he has spent countless time in shaping me into a better researcher. A PhD journey consists of many ups and downs. However, since I met him, I have been always running "gradient ascent" with the right step size. It is simply impossible to fully describe how great Devavrat is. I really cannot thank him enough given the influence he had on me both as an academic advisor and as a friend. I certainly would not have been who I am today without his generous guidance. It is such an honor and blessing to have met him in my life, and I very much look forward to the new chapter of our friendship after graduation.

I am also grateful to my committee members, John Tsitsiklis and Leslie Kaelbling. John was also my advisor for the Master's thesis. He is not only knowledgeable in various fields, but could also always see the very essence of the seemingly messy problems. Our meetings have always been a valuable source of inspiration for me. The first machine learning course I took at MIT is 6.867 in Fall 2016 when Leslie was one of the instructors. Undoubtedly, it was enlightening and triggered my interest for future research in this area.

Further, I would like to thank my wonderful collaborators over the years. This thesis would not have been possible without them, especially Qiaomin Xie, Dogyoon Song, and Yuzhe Yang. I have known Qiaomin since my undergraduate studies, and it is absolutely

amazing to formally collaborate with her on a variety of problems after she joined Devavrat's group as a post-doc. In that sense, I am also her first graduate student. I am glad that through our collaborations, I finally made Dogyoon a bit more interested in reinforcement learning. Yuzhe and I met each other initially on a deep learning project. Since then, I have very much enjoyed our discussions on many innovative ideas, some of which indeed led to great publications. I have learned much from him and hopefully, I have been helpful for his early career at MIT as well. Our collaborations on several applied machine learning problems nicely complemented my theoretical research and have made me a more comprehensive researcher. I will always remember those nights we spent together right before the deadlines and I am certain that he will have a bright future at MIT.

In addition, there are many friends at MIT to thank. I have been living with Guo Zhang for five years, almost my entire MIT journey. In fact, I met my wife during a Thanksgiving event he organized. I am glad that I finally collaborated with Anish Agarwal and Dennis Shen before my graduation. I will miss the time we spent together in our office, 32D-666. I enjoyed many fruitful conversations with Chengtao Li, Ruihao Zhu, and Sarah Cen. I would also like to thank Quan Li, Kevin Li, Jianan Zhang, Xinzhe Fu, and Hao He for discussions and help during different stages of my internship and full-time job search. More broadly, I thank the friends at 32D-666, SPPIN, and LIDS/EECS for making my stay at MIT, particularly at Stata Center, memorable. I cherish the conversations with all of you and sincerely wish that our friendship will continue beyond MIT.

For my graduate studies, I was supported by MIT Jacobs Presidential Fellowship, Siemens Futuremakers Fellowship, as well as grants from KACST (grant agreement dated 07/01/2011) and NSF (CMMI-1634259). My research would not have been smooth without their financial support.

Last but not least, I am indebted to my family for their unconditional support and love. This thesis is dedicated to them. My parents, Meiqin Su and Bifeng Xu, basically have no idea of my research. Yet, they try to understand and support me in their own way, which is indeed important to me. Thank you for believing in me and providing me with the best educational resources you could since I was young. I met my wife, Chengcheng Qin, while she was a student at Harvard. I very much cherish the classes we took together at Harvard as well as the time we spent in Cambridge. Since our marriage in 2017, she has been working alone in New York City to support my PhD dream. I cannot imagine how much she has sacrificed for the family over the years. Thank you for loving me, encouraging me, and taking care of me. Undoubtedly, I would not have gone so far without her love, support, and sacrifice. I owe my deepest gratitude to her.

# Contents

# List of Figures

# List of Tables

Chapter 1

# Introduction

Sequential decision-making is ubiquitous in a variety of domains in the real-world. In healthcare, treatment plans are applied sequentially and revised adaptively according to the progress and the observed health state of the patients. In robotic tasks such as locomotion and grasping, a sequence of appropriate actions that manipulates the robot must be executed sequentially in order to achieve the desired behavior. In personal investment, we make decisions to allocate the wealth to different assets from time to time, adjusting them so as to maximize the long-term wealth. A prominent feature of all the aforementioned examples is the long-term dependence: instead of just an immediate reward following an action, the decision maker seeks a sequence of actions that would maximize the cumulative long-term outcomes such as the eventual health of the patients or the eventual wealth after retirement.

Reinforcement Learning (RL) has recently emerged as a generic yet promising technique for sequential decision-making tasks. Typical settings of those tasks, such as the ones above, involve a decision maker interacting with the environment. Decisions (potentially sub-optimal) are attempted, with corresponding feedback from the environment received back by the decision maker. RL is precisely about learning from those interactions to seek an optimal policy, through the formal framework of Markov Decision Processes (MDP). In general, it has a rich history [158, 22, 82], but recent advancements in machine learning, optimization, statistics and importantly computational resources have truly pushed the boundary of modern RL techniques. RL, coupled with expressive function approximators such as neural networks, has been the main technique underpinning remarkable successes in a variety of domains, such as game playing [122, 150, 149], robotics [108, 86], recommendation system [35, 159], autonomous driving [136, 91] and healthcare [171, 127].

This thesis aims to further our understanding about modern RL algorithms. In particular, we will focus our investigation on the *data efficiency* through a *"structural"* lens, which will be made clearer throughout this thesis.

## 1.1 Reinforcement Learning

We begin with a short introduction to Reinforcement Learning, including its generic setting and common notation used in this thesis, to setup the necessary background.

RL considers the problem of learning through agent-environment interaction. At each discrete time step $t$, an agent (or decision maker) begins at a state $s$ and then decides an action $a$ to take. Upon receiving the action, the environment transits to a next state $s'$ via certain rules associated with the current state-action pair, $(s, a)$. The environment also returns a corresponding immediate reward for the state-action pair. The process is then repeated at the next time step $t + 1$, and the agent's goal is to maximize the cumulative reward s/he could possibly get. In the previous example of personal investment, the state $s$ could be the distribution of wealth on different assets at the current time; the action $a$ could be to buy or sell some portions of the assets; the immediate reward could be the corresponding cash outflow or inflow; the distribution of wealth then changes as a result of the previous distribution and the buy/sell action.

Formally, the above setup can be described as an infinite-horizon discounted Markov Decision Process (MDP), which includes five components, $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. $\mathcal{S}$ and $\mathcal{A}$ are the state space and the action space, respectively. In this thesis, we generally consider continuous state space. $\mathcal{P}(s'|s, a)$ is the unknown transition kernel which determines the probability of transitioning to state $s' \in \mathcal{S}$ given the current state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$. $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function which determines the immediate reward (possibly random) received for taking action $a \in \mathcal{A}$ at state $s \in \mathcal{S}$. Finally, $\gamma \in (0, 1)$ is the discounting factor. A policy $\pi(a|s)$ specifies the probability of selecting action $a \in \mathcal{A}$ at state $s \in \mathcal{S}$. The standard value function associated with a policy $\pi$ is defined as

$$V^\pi(s) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t) \mid s_0 = s\Big],$$

where $a_t \sim \pi(\cdot|s_t)$ and the expectation is taken with respect to the randomness of the state transition, the policy and the reward function (if random). That is, $V^\pi(s)$ is the discounted cumulative reward obtained by starting at state $s$ and following the policy $\pi$ to choose actions at each time step. The optimal value function, denoted by $V^*$, is the value function of the reward-maximizing policy, i.e.,

$$V^*(s) = \sup_\pi V^\pi(s), \ \forall\, s \in \mathcal{S},$$

and the corresponding reward-maximizing policy is commonly referred as the optimal policy, denoted by $\pi^*$. It is well understood that such an optimal policy exits in reasonable generality [23]. Another "value function" that is critical in designing several RL algorithms

is the state-action value function, or $Q$-function:

$$Q^\pi(s,a) = \mathbb{E}[\mathcal{R}(s,a)] + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)}[V^\pi(s')].$$

In other words, $Q^\pi(s,a)$ is the discounted cumulative reward by starting at state $s$, taking action $a$ immediately and then following the policy $\pi$ afterwards. Similarly, we define the optimal $Q$-function, denoted by $Q^*$, as

$$Q^*(s,a) = \mathbb{E}[\mathcal{R}(s,a)] + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)}[V^*(s')].$$

Generically speaking, RL concerns about either learning the optimal values, $V^*$ or $Q^*$, or learning the optimal policy, $\pi^*$. Throughout, to measure the data efficiency of an algorithm, we use the standard asymptotic notation, $O(\cdot)$ and $\Omega(\cdot)$, to quantify the number of samples involved. We also use $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ to hide logarithmic factors for simplicity.

**Terminology.** Literature on RL is rich and is growing rapidly. In what follows, we provide a concise summary of the various classes of RL algorithms and the related terminologies, with the aim to provide better context for this thesis.

*Model-free versus Model-based:* as the name suggests, model-free methods learn the quantity of interest directly without learning a model of the environment (i.e., the transition kernel). A classical example is the $Q$-learning algorithm which iteratively updates the $Q$-value through temporal difference estimates [175, 122]. In contrast, for model-based RL methods, the transition dynamics is learnt and subsequently utilized for policy learning (e.g., direct planning through Model Predictive Control [61, 29] or data augmentation for model-free methods) [173, 139, 78, 172, 83, 113, 51, 107, 126, 68]. Thus far, model-free methods are generally more prevalent and more thoroughly developed due to their flexibility and ease of application [122, 121, 180, 140, 141, 110, 69, 59, 174, 75]. Learning a model accurately is fundamentally hard, and the bias can lead to poor policy learning. However, recent work has demonstrated that, when designed appropriately, model-based approaches can be far more data-efficient in terms of the number of samples required to learn a good policy and can also generalize as well as the state-of-the-art model-free methods [41, 42, 102, 83, 70]. Designing efficient RL methods, possibly through learning models, is an active research area.

*Value-based versus Policy Optimization:* recall that the agent's goal is to maximize the cumulative reward. Policy optimization achieves so by directly parameterizing the policy (i.e., $\pi_\theta(a|s)$ with parameter $\theta$) and then optimizing it through gradient steps with respect to objectives that are either the cumulative reward or some forms of approximations [176, 140, 141, 85]. In comparison, value-based methods do so indirectly by learning the value functions, in particular, the optimal $Q$-value. Methods in this class often parameterizes the $Q$-value (i.e., $Q_\theta(s,a)$) and typically optimizes it with respect to objective functions based

on the Bellman equation [122, 75, 174, 167, 180, 8]. In addition, policy optimization methods are almost always updated in an on-policy manner, meaning that at each update step $k$, the data used must be generated according to the current policy $\pi_{\theta_k}$. Value-based methods, on the other hand, are mostly off-policy methods: they can utilize all the historical data, regardless of the underlying policies that generate them. Indeed, modern methods often include an experience reply buffer [122, 8] to effectively leverage such off-policy property. It is worth mentioning that the distinction of value-based or policy optimization methods is not strict. In fact, there is a variety of algorithms in between, often under the name of actor-critic methods [98, 110, 69, 59]. They parameterize both the policy (i.e., the actor) and the value function (i.e., the critic) with the aim of improving the learning process from the help of each other. Further, modern implementation of some policy optimization methods also involves a parameterized value function, serving as a baseline to reduce the variance in the associated gradient estimates.

*Online versus Offline:* thus far, the vast majority of RL literature has focused on the "online" setting, where the agent is able to continuously and adaptively sample transition data during the learning process. Such online sampling is instructive in guiding the overall learning process, helping these methods effectively quantify and reduce uncertainty for unseen state-action pairs. However, this type of interaction with the environment can be costly or simply infeasible for numerous real-world applications such as healthcare, autonomous driving, and socio-economic systems. As a result, there has been a rapidly growing literature on "offline RL" (or batch RL) [109, 100, 112, 60, 105, 177, 5, 101, 183, 93], which focuses on leveraging pre-collected, fixed data to learn the RL quantity of interest. Naively, many (online) off-policy methods [69, 59, 110, 66, 122] can be directly applied in this offline setting, as they are in principle designed to leverage historical data and to be independent of the data generating policies. Their performance, however, degrades significantly [60, 100]. Without the ability to perform online correction, these methods suffer severely from distributional shift [109] and simply fail most of the time. To tackle this challenge, the current literature often designs policies that are "close", in an appropriate sense, to the observed behavioural policy in the offline dataset, via directly quantifying and regularizing the uncertainty for a given state-action pair [100, 177, 60].

**This Thesis.** This thesis will explore both value and policy learning, in a model-free manner. Following literature on theoretical reinforcement learning, we consider the setup with the access of a generative model (i.e., a simulator) [84]. This means that the transition kernel $\mathcal{P}$ and the reward function $\mathcal{R}(s,a)$ are unknown, but the agent could access the transition data through querying the generative model at any state-action pair $(s,a)$. The complexity (or data efficiency) of an algorithm is then measured through the number of transition data required to achieve a certain guarantee. Naturally, we consider the online

setup where we are able to query the generative model at will throughout the process. After we develop a better understanding of the results, potential extensions and connections to other types of settings shall be discussed at the end of the thesis.

## ■ 1.2 Theme of The Thesis

With remarkable advancements in computational resources, the successes of many modern RL algorithms are often centered around the usage of huge amount of data and the cleverness in how they are leveraged. For instance, on Atari games, Deep Q-learning requires 50 millions training frames (around 38 days of game experience in total) and meanwhile, utilizes a reply buffer of 1 million frames to cleverly leverage past transition data to update and regularize the neural network [122]. In practice, different applications naturally admit different data patterns: some, such as video games and robotics, may be easy to request more data while others like healthcare may be costly or infeasible to do so. As RL is becoming increasingly popular and widely used across various domains, it is vital to develop a better understanding about the usage of data involved in its algorithmic design. Therefore, this thesis focuses on investigating the *data efficiency* of RL. This naturally leads to two quests: understanding and improvement. More precisely,

1. Can we characterize the data complexity for existing, empirically successful algorithms?
2. Further, can we design provably data-efficient RL algorithms?

This thesis will take a "structural" perspective to explore the above questions. To be successful or efficient, RL algorithms often utilize certain kinds of structures within the decision-making tasks. At a high level, this structure can be: (1) a different representation/viewpoint of the underlying problem, such as representing the next-steps transitions into trees and subsequently producing value estimates in the Monte Carlo Tree Search; (2) certain mathematical properties of the task, such as having Lipschitz or "low-rank" value functions or admitting a "stable" policy. While the term "structure" may be vague at this moment, we will unpack this perspective and its usefulness throughout the thesis. To achieve this, the thesis is divided into three parts. In the following, we give a preview of each part.

### ■ 1.2.1 Monte Carlo Tree Search

Arguably, one of the breakthroughs of RL in recent years is AlphaGo [148]. Go is traditionally considered as the most challenging classical games in artificial intelligence. Remarkably, AlphaGo defeated even world champions, setting new standards of superhuman performance. Such phenomenal success naturally motivates the investigation on its algorithm.

In particular, what is the data efficiency involved that leads to the superior performance empirically?

At a high level, the overall learning algorithm of AlphaGo Zero (AGZ) [150], a successive, improved version of AlphaGo, can be summarized as an effective combination of Monte Carlo Tree Search (MCTS) and supervised learning. The algorithm performs a kind of approximate dynamic programming with a simulator. It employs supervised learning to learn a policy/value function (represented by a neural network) based on samples generated via MCTS; the neural network is recursively used to estimate the value of leaf nodes in the next iteration of MCTS for simulation guidance, and the eventual sample data generated by the MCTS-based policy is applied for gradient update of the network parameters. In the first part of this thesis, we focus our investigation on the non-asymptotic behavior of MCTS, as it is the key component behind AGZ's success. This will also enable us to subsequently analyze the overall data complexity when it is combined with supervised learning for iteratively improving the estimation of value function.

As the name suggests, the key structure behind the power of MCTS is the tree representation. By viewing the multi-step transitions as trees, one can utilize the representation to design effective algorithms to evaluate the value of the current state by drawing insights from the bandit literature. Let us now elaborate this aspect. Consider for a moment that we only attempt to maximize a one-step reward, i.e., we start with the initial state $s_0$ and the process terminates after we take the first action $a_1$. This is commonly referred as the Multi-armed Bandit (MAB) problem [6, 13, 104, 27], where the goal is to discover amongst finitely many actions (or arms), the one with the best average reward while choosing as few non-optimal actions as possible during the learning process. The rewards for any given arm are assumed to be independent and identically distributed (i.i.d.). A common challenge in such setting is the exploration-exploitation tradeoff: should we choose the action that has the highest empirical reward so far or should we explore more options (potentially sub-optimal) so as to discover the reward distribution better. The classical Upper Confidence Bound (UCB) algorithm resolves the tradeoff by utilizing the exponential concentration for such i.i.d. and hence stationary reward processes at each arm: at each time of the learning process, choose action with the maximal index (ties broken arbitrarily), where the index of an arm is defined as the empirical mean reward plus a bonus term which is a constant times $\sqrt{\log t/s}$. Here, $t$ is the total number of trials so far and $s \leq t$ is the number of times the particular action is chosen in these $t$ trials. Note that the bonus depends *logarithmically* on $t$.

The goal of MCTS is very similar to the MAB setup described above – choose an action at a given query state that gives the best average reward. However, instead of only a one-step immediate reward, the overall reward in general depends on the future actions.

Therefore, to determine the best action for the given state, one has to take future actions into account, and MCTS does this by simulating future via effectively expanding all possible future actions recursively in the form of trees. In essence, the optimal action at the root of such a tree is determined by finding optimal path in the tree. And determining this optimal path requires solving multiple MABs, one per each intermediate node within the tree. Apart from the MABs associated with the lowest layer of the tree, all the MABs associated with the intermediate nodes turn out to have rewards that are the rewards generated by MAB algorithms for nodes downstream. This creates complicated, hierarchically inter-dependent MABs. Even though, because of the similarity, it is still natural and tempting to directly apply the UCB algorithm for MAB to this hierarchical MAB on trees. That is, at each node of the tree, one applies the UCB algorithm with a *logarithmic* bonus to choose an action during the simulation. This in fact leads to one of the earliest and most popular form of MCTS, the UCT (Upper Confidence Bounds for Trees) algorithm [94, 95]. In [94, 95], certain asymptotic optimality property of UCT is claimed. The proof therein is, however, incomplete, as we discuss in greater details in Chapter 2.

Indeed, to determine the appropriate, UCB-like index algorithm for each node of the MCTS tree, it is essential to understand the concentration property of the rewards, i.e., concentration of regret for MABs associated with nodes downstream. In standard MAB, the logarithmic bonus term originates from the exponential concentration given the i.i.d rewards. In the MCTS tree, while the rewards at leaf level may enjoy exponential concentration due to independence, the regret of any algorithm even for such a MAB is unlikely to have exponential concentration in general [12, 137]. Further, the MAB of our interest has non-stationary rewards due to strong dependence across hierarchy. Indeed, an oversight of this complication led [94, 95] to suggest UCT inspired by the standard UCB algorithm for MABs with stationary, independent rewards.

The goal of the first part of this thesis is hence to provide a rigorous theoretical foundation for MCTS with the correct concentration properties. We will show that the correct concentration should be *polynomial* instead of exponential. The analysis and the result can serve as an example for how one could extend and analyze a traditional MAB algorithm to the multi-step tree setup. This opens new directions such as adapting continuous armed bandit algorithm to MCTS with continuous domains [117]. Once this non-asymptotic guarantee of MCTS is established, we can then readily understand the data complexity of AlphaGo-style algorithm that combines MCTS with supervised learning. To summarize, we are interested in the following questions:

- What is the appropriate form of MCTS for which the asymptotic convergence property claimed in the literature [94, 95] holds?

- Can we rigorously establish the "strong policy improvement" property of MCTS when combined with supervised learning as observed in the literature (e.g., in [150])? If yes, what is the quantitative form of it?

- Does supervised learning combined with MCTS lead to the optimal policy, asymptotically? If so, what is its finite-sample (non-asymptotic) performance?

### ■ 1.2.2 Data-efficient "Low-rank" RL

Having analyzed the data efficiency of empirically successful algorithms, the natural next step is to consider how we could further improve the algorithmic designs of simulation-based RL methods. We start with compact domains in the second part of the thesis and focus on learning the optimal $Q$-value, $Q^*$.

Generic learning methods often suffer from "curse-of-dimensionality": the sample complexity for learning any quantity of interest often scales exponentially in the dimension. Specifically, in RL, by viewing the problem of learning $Q^*$ as a non-parametric regression problem, the classical minimax theory [154, 164] suggests that for $\varepsilon > 0$, we need $\Omega(\varepsilon^{-(d_1+d_2+2)})$ samples to learn an $\varepsilon$-optimal $Q$-function, when the (continuous) state and action spaces have dimensions $d_1$ and $d_2$ respectively and the $Q$-function is Lipschitz continuous over them. Here, an $\varepsilon$-optimal $Q$-function means an estimate $\hat{Q}$ that is uniformly close to $Q^*$ over the state-action domain, i.e.,

$$\sup_{s\in\mathcal{S},a\in\mathcal{A}} |\hat{Q}(s,a) - Q^*(s,a)| \le \varepsilon.$$

While this worst-case result is discouraging, practical RL tasks, as exemplified by empirical successes, seem to possess low-dimensional latent structures that make learning with limited data possible in the real world. Indeed, feature-based literature precisely aims to explain such phenomenon by positing that either the transition kernel [178, 179] or the value function [163, 119, 129, 114, 184] is "structured", often linear in low-dimensional features associated with states and actions. For example, [178] considers feature-based linear transition model where the transition kernel $\mathcal{P}(s'|s,a) = \sum_{k\in[K]} \phi_k(s,a)\psi_k(s')$ for a known feature map $\phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^K$ and some unknown functions $\psi_1, \psi_2, \ldots, \psi_K : \mathcal{S} \to \mathbb{R}$. Such structural knowledge then enables the design of an efficient $Q$-learning algorithm whose sample complexity scales with the feature dimension $K$ instead of the dimension of the state space and/or the action space. In general, while feature-based structures may be true, the algorithm may not have the *knowledge* of such feature map beforehand in practice; and relying on the hope of a neural network to find it might be too much to ask.

Motivated by this, the primary goal in this part of the thesis is to learn the optimal

$Q$-function in a data-efficient manner if it has a lower-dimensional representation, *without* the need of any additional information such as knowledge of features. Therefore, we ask the following key questions:

- Is there a universal representation of $Q$-function that allows for designing a data-efficient learning algorithm if the $Q$-function has a low-dimensional structure?
- If so, how to leverage the structure in a principled manner?

We will answer these in the affirmative by first developing a spectral representation of the $Q$-function for a generic RL task. Under mild technical conditions, we show that $Q^*$ admits a general representation:

$$Q^*(s,a) = \sum_{i=1}^{\infty} \sigma_i f_i(s) g_i(a), \quad \forall\ s \in \mathcal{S}, a \in \mathcal{A}, \tag{1.1}$$

with $\sum_{i=1}^{\infty} \sigma_i^2 < \infty$, and $\{f_i : i \in \mathbb{N}\}$ and $\{g_i : i \in \mathbb{N}\}$ being orthonormal sets of functions. That is, for any $\delta > 0$, there exists $r(\delta)$ such that the $r(\delta)$ components in (1.1) provide $\delta$-approximation of $Q^*$. This inspires a parametric family of $Q^*$ parameterized by $r \geq 1$, i.e., $Q^*(s,a) = \sum_{i=1}^{r} \sigma_i f_i(s) g_i(a)$, with all Lipschitz $Q^*$ captured as $r \to \infty$. When $r$ is small, it suggests a form of lower-dimensional structure within $Q^*$: we call such a $Q^*$ to have (low) *rank* $r$.

Given the above universal representation with the notion of dimensionality for $Q^*$ through its rank, we focus on leveraging the structure to learn $Q^*$ efficiently when $Q^*$ is of low rank $r$. To achieve this, let us take a matrix viewpoint of the $Q$-function to develop the intuition. Notice that for any set of $m$ states $\{s_k\}_{k=1}^m$ and $n$ actions $\{a_\ell\}_{\ell=1}^n$, the induced matrix $[Q^*(s_k, a_\ell) : k \in [m], \ell \in [n]]$ has rank (at most) $r$. Naively, when the $m$ chosen states "cover" $\mathcal{S}$ finely ($n$ actions cover $\mathcal{A}$, respectively) and suppose we also have a good estimate for the entire matrix, we can estimate $Q^*$ for the entire domain $\mathcal{S} \times \mathcal{A}$ by interpolating the estimates for the $mn$ entries. This leads to the sample complexity of $\tilde{O}\big(\varepsilon^{-(d_1+d_2+2)}\big)$, matching the mini-max lower bound. On the other hand, this also suggests that in order to further improve the sample complexity, we should aim to produce good estimate without estimating the whole matrix. In other words, if we are able to get a good estimate of the entire matrix by only estimating a selective portion of the entries instead of all the $mn$ entries, then such improvement will be directly translated to a reduced sample complexity.

The above intuition motivates the design of an appropriate, efficient matrix estimation technique, and this is precisely where the low-rank structure of $Q^*$ becomes crucial. In literature, matrix estimation concerns completing a $m \times n$ matrix from partial, noisy observation of it. This problem has been extremely well studied [132, 31, 32, 97, 34, 38, 49, 37] with provable recovery guarantees for low-rank matrices. However, most recovery guarantees are

given in terms of Frobenius norm of the error, or mean squared error. In our case, note that the completed matrix in fact contains operational meaning: each entry is an estimate of $Q^*(s_k, a_l)$, $k \in [m]$ and $l \in [n]$. Therefore, in order to obtain an $\varepsilon$-optimal $Q$-function (in the $\ell_\infty$ sense) from the completed matrix, we need the matrix estimation technique to reliably estimate each entry, i.e., with $\ell_\infty$ error guarantee. This is technically hard and there are only limited results [52, 39]. To make matters worse, because an accurate estimate of $Q^*(s, a)$ necessarily depends on the infinite future, the measurement noise for estimating entries of the matrix in our setting can be unavoidably arbitrary (not necessarily zero mean) though bounded. Therefore, a new method is required and that is precisely what we will achieve.

To summarize, using the matrix estimation technique introduced and starting with 0 as initial estimate of $Q^*$, our estimate is improved iteratively by inter-leaving one-step look-ahead (for estimating a portion of the entries) and matrix estimation (for completing the rest of the matrix). Such approach of leveraging the low-rank structure of $Q^*$ eventually leads to a data-efficient RL algorithm whose sample complexity only scales as $\tilde{O}\big(\varepsilon^{-(\max\{d_1,d_2\}+2)}\big)$.

### ■ 1.2.3 Stability in Unbounded State Space

In the last part of this thesis, we continue our investigation on data-efficient RL, but focusing on unknown dynamical systems with an unbounded state space. Such problems are ubiquitous in various application domains, as exemplified by scheduling for networked systems. While algorithms for the setting with finite, bounded or compact state spaces have been well studied, with both classical asymptotic results and recent non-asymptotic performance guarantees, literature on problems with unbounded state space is scarce, with few exceptions such as linear quadratic regulator [1, 50], where the structure of the dynamics is *known*. Indeed, the unboundedness of the state space presents with new challenges for algorithm or policy design, as well as analysis of policy in terms of quantifying the "goodness". As we will develop, the structure that helps to systematically explore and resolve those challenges is the stability of the system.

First, to exemplify the challenges involved, let us consider a simple example of discrete-time queueing system with two queues as shown in Figure 1.1. Jobs arrive to queue $i \in \{1, 2\}$ per Bernoulli process with rate $\lambda_i \in (0, 1)$. A central server can choose to serve job from one of the queues at each time, and if a job from queue $i$ is chosen to serve, it departs the system with probability $\mu_i \in (0, 1)$. That is, in effect $\rho_i = \lambda_i/\mu_i$ amount of "work" arrives to queue $i$ while total amount of work the system can do is 1. The state of the system is $q = (q_1, q_2)$ with $q_i$ representing number of jobs in the $i$th queue. The evolution of the system is controlled by a scheduling decision that specifies which queue $i \in \{1, 2\}$ to serve at each time. Viewed as a MDP, the state space is $\mathcal{S} = \{0, 1, \ldots\} \times \{0, 1, \ldots\}$ and the action

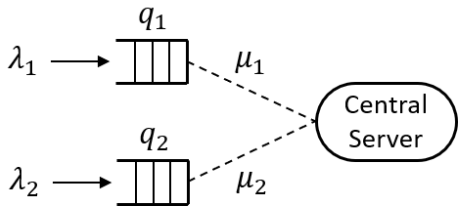



**Figure 1.1.** A two queue network: arrival, service rates for queue $i \in \{1, 2\}$ are $\lambda_i$, $\mu_i$, respectively.

space is $\mathcal{A} = \{1, 2\}$. Operated under policy $\pi$, the server will serve queue $\pi(q)$ at state $q$. The problem of stochastic control of the network is to identify a policy that optimizes a given criterion (e.g., average or discounted total queue lengths).

In traditional RL approaches, the policy is trained *offline* using finitely many samples for finite, bounded or compact state spaces and then it is deployed in *wild* without further changes. Here, the term "offline" should not be confused with the offline RL setting introduced in Section 1.1. We use offline here to distinguish the stages between learning and deployment: for a real system, one can train a policy in the hindsight with finite samples and then deploy without changes when running the system (i.e., offline) or continuously train/update the policy while it is running (i.e., online). A natural adaption of the traditional RL approaches is by restricting the RL policy to a finite subset of the state space chosen appropriately or arbitrarily for training. However, even in our simple, motivating example, the system will reach a state $q$ not contained in the finite training data with non-zero probability. Due to unboundedness, such a state $q$ can be arbitrarily far from any training data, and hence the estimate for $q$'s transition probabilities and value function will remain at their initial/default values (say 0) or will be highly inaccurate. With such an uninformative estimate, the corresponding policy will be independent of the state $q$ or simply erroneous. And it is likely that the policy may end up serving empty queue with a nonzero probability. This might cause the queues to grow unboundedly with strictly positive probability. Clearly, more sophisticated approaches to truncate systems are not going to help as they will suffer from a similar issue.

An alternative to truncation is to "compactify" the state space by mapping the unbounded space to a bounded set. The problem is then reduced to one with a bounded domain. However, traditional RL approaches may also fall short for the reduced problem: properties of the original problem that allow for efficient learning can be easily destroyed under the mapping. Consider a simple example where the state space is $\mathcal{S} = (-\infty, \infty)$. For continuous problems, certain smoothness property is necessary for efficient learning. Suppose that for the original MDP, the optimal value function $V^*$ that we wish to learn is $\zeta$-Lipschitz for a constant $\zeta > 0$. Consider a natural mapping $z = \tanh s$, which "compactifies" the unbounded space $\mathcal{S}$ to $[-1, 1]$. Using chain rule, we have $|\frac{\partial V^*}{\partial z}| = |\frac{\partial V^*}{\partial s} \cdot \frac{\partial s}{\partial z}| \leq \zeta \cdot |\frac{1}{1-z^2}|$. As the

original state $s$ approaches infinity, $z$ approaches either 1 or $-1$, in which case the derivative becomes infinity, implying that the smoothness property is completely lost. Therefore, it is impossible to learn the function over the bounded set well with *finite* samples. Such issues will be exaggerated in higher dimensions. In general, this kind of state space compactification suffers similar issues as truncation: it necessarily discounts/skews large states, which are exactly the states we care about when studying stability. It seems extremely challenging to find a proper mapping that preserves all the nice properties; thus efficient learning in the "compactified" space is far from obvious if not impossible. Finally, another potential approach is to find "lower-dimensional structure" through functional approximation, e.g., by parametrizing the policy $\pi$ within some function classes (such as linear functions or neural networks). For this approach to work, the function class must be expressive enough to contain a stable policy. However, it is not at all clear, *a priori*, which parametric function class has this property, even for the simple example in Figure 1.1. This challenge is only exacerbated in more complicated systems. Although some approximation architectures work well empirically [116, 115, 47], there is no rigorous performance guarantee in general.

To sum up, the traditional RL approaches for finite, bounded or compact state space are not well suited for systems with unbounded state space. Approaches that rely on *offline* training only are bound to fail as system will reach a state that is not observed in finitely many samples during offline training and hence, there is no meaningful guidance from the policy. Therefore, to learn a reasonable policy with an unbounded state space, the policy ought to be updated whenever a new scenario is encountered. That is, we need to consider *online* policies, i.e., one that is continuously updated upon incurring new scenarios. Another challenge is in analyzing or quantifying "goodness" of such a policy. Traditionally, the "goodness" of an RL policy is measured in terms of the error induced in approximating, for example, the optimal value function over the entire state space; usually measured through $\|\cdot\|_\infty$ norm error bound [90, 23]. Since the state space is unbounded, expecting a good approximation of the optimal value function over the entire state space is not a meaningful measure. Therefore, we need an alternative to quantify the "goodness" of a policy. Overall, these motivate the investigation of the following questions:

1. What is the appropriate "goodness" of performance for a RL policy for unbounded state space?

2. Is there an online, data-driven RL policy that achieves such "goodness"? And if so, how does the number of samples required per time-step scale?

Inspired by queueing network and control theory, we introduce a notion of stochastic stability to quantify the "goodness" of a policy. Indeed, when facing systems with unbounded domains, it is often desired to operate the system under a small, bounded regime and to

be able to guide the system back to the regime when it occasionally leaves it. Back to our queueing example, this means a queueing network with small queue lengths and hence, smaller system delay. We introduce a formal definition of stability to quantify the above requirements. With the framework at hand, we then investigate how to obtain an online, efficient stable policy. The structure that enables this is the stability of the unknown optimal policy. Arguably, real system of interest ought to be stable under the policy with maximum reward. Indeed, operating under a small, bounded regime (and hence stable) is often highly correlated with the system performance, as exemplified in our queueing example. We hence consider systems with such a structure and will demonstrate an efficient, stable policy that also automatically adjusts its hyper-parameters based on formal statistical tests.

## ■ 1.3 Contributions

Through the studies of MCTS, low-rank $Q^*$ and stability, this thesis makes a step forward in understanding the data efficiency of RL by investigating the two quests posted in Section 1.2. The structures within each problem enable such analysis and improvement. Below, we summarize the methodological, algorithmic and technical contributions of this thesis.

### ■ 1.3.1 Methodological Contributions

**Non-stationary MAB and Recursive Polynomial Concentration (Chapter 2).** To overcome the challenges in analyzing the hierarchical dependence of the MCTS tree, we formulate an appropriate form of non-stationary MAB which correctly models the MAB at each of the node in the tree. For such a non-stationary MAB, we define UCB algorithm with appropriate index and under which we establish appropriate concentration of the induced regret. This, in turn, allows us to recursively define the UCT algorithm for MCTS, from the leaf level to the root level, by appropriately defining index for each of the node-action within the MCTS tree. By analyzing the concentration of each level, we conclude the desired concentration property at the root node, i.e., the output of MCTS.

**Low-rank Representation (Chapter 3).** Given state space $\mathcal{S} = [0,1]^{d_1}$ and action space $\mathcal{A} = [0,1]^{d_2}$, let $Q^* : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be the optimal $Q$-function for the RL task of interest. We consider the integral operator $K = K_{Q^*}$ induced by $Q^*$ that maps an integrable function $h : \mathcal{S} \to \mathbb{R}$ to $Kh : \mathcal{A} \to \mathbb{R}$ such that $Kh(a) = \int_{s\in\mathcal{S}} Q^*(s,a)h(s)ds$, $\forall a \in \mathcal{A}$. For Lipschitz $Q^*$, we show that $K$ is a Hilbert-Schmidt operator admitting generalized singular value decomposition. This leads to the spectral representation of $Q^*$: $Q^*(s,a) = \sum_{i=1}^{\infty} \sigma_i f_i(s) g_i(a)$, $\forall\ s \in \mathcal{S}, a \in \mathcal{A}$. As mentioned, this inspires a parametric family of $Q^*$ parameterized by $r \geq 1$, i.e., $Q^*(s,a) = \sum_{i=1}^{r} \sigma_i f_i(s) g_i(a)$, with all Lipschitz $Q^*$ captured as $r \to \infty$. Importantly, when $r$ is small (i.e., $Q^*$ is low-rank), we introduce in Matrix

Estimation (ME) technique to "complete" estimation of $Q^*$ based on estimates for only a small, selective portion of the state-action space. Such usage of ME leads to a significant improvement on data efficiency, but has not been explored in RL before. To the best of our knowledge, this is the first work to show a provable, quantitative utility of exploiting the low-rank structure to reduce sample complexity in $Q$-learning. Moreover, the main methodology we develop remains valid and applicable to various problems in machine learning beyond RL, which involve a bi-variate function possessing a low-rank structure.

**Notion of Stability (Chapter 4).** For systems with unbounded state space, we introduce a notion of *stability* to quantify "goodness" of RL policies for unbounded state space inspired by the literature in queueing systems and control theory. Informally, a RL policy is stable if the system dynamics under the policy returns to a finite, bounded or compact subset of the system infinitely often — in the context of our queueing example, it would imply that queue sizes remain finite with probability 1. For applications where instability implies unbounded cost, the notion of stability provides a meaningful notion of first-order optimality; Indeed, further refined notions of performance beyond stability, such as diffusion-approximation or heavy traffic analysis as typically considered in queueing systems would be natural next steps to consider.

### ■ 1.3.2 Algorithmic & Technical Contributions

**Corrected UCT for MCTS and Non-asymptotic Analysis (Chapter 2).** We consider the following non-stationary MAB with a polynomial concentration, which serves as the key building block in analyzing MCTS. Given $[K] = \{1, \ldots, K\}$ actions or arms, let $X_{i,t}$ denote the reward generated by playing arm $i \in [K]$ for the $t$-th time. Let empirical mean over $n$ trials for arm $i$ be $\bar{X}_{i,n} = \frac{1}{n}\sum_{t=1}^{n} X_{i,t}$, and let $\mu_{i,n} = \mathbb{E}[\bar{X}_{i,n}]$ be its expectation. Suppose $\mu_{i,n} \to \mu_i$ as $n \to \infty$ for all $i \in [K]$ and let there exist constants, $\beta > 1$, $\xi > 0$, and $1/2 \leq \eta < 1$ such that for every $z \geq 1$ and every integer $n \geq 1$,

$$\mathbb{P}\big(|n\bar{X}_{i,n} - n\mu_i| \geq n^{\eta} z\big) \leq \frac{\beta}{z^{\xi}}.$$

Note that for i.i.d. bounded rewards, above holds for $\eta = 1/2$ for any finite $\xi$ due to exponential concentration. We propose to utilize the UCB algorithm where at time $t$, the arm $I_t$ is chosen according to

$$I_t \in \arg\max_{i \in [K]} \big\{\bar{X}_{i,T_i(t-1)} + B_{t-1,T_i(t-1)}\big\}, \tag{1.2}$$

where $T_i(t) = \sum_{l=1}^{t} \mathbb{I}\{I_l = i\}$ is the number of times arm $i$ has been played, up to (including) time $t$, and the bias or bonus term $B_{t,s}$ is defined as $B_{t,s} = \frac{\beta^{1/\xi} \cdot t^{\eta(1-\eta)}}{s^{1-\eta}}$. Let $\mu_* = \max_{i \in [K]} \mu_i$

and let $\bar{X}_n$ denote the empirical average of the rewards collected. Then, we establish that $\mathbb{E}[\bar{X}_n]$ converges to $\mu_*$, and that for every $n \geq 1$ and every $z \geq 1$, a similar polynomial concentration holds:

$$\mathbb{P}\big(|n\bar{X}_n - n\mu_*| \geq n^{\eta} z\big) \leq \frac{\beta'}{z^{\xi'}},$$

where $\xi' = \xi\eta(1-\eta) - 1$, and $\beta' > 1$ is a large enough constant.

For MCTS, as discussed, the leaf nodes have rewards that can be viewed as generated per standard stationary MAB. Therefore, the rewards for each arm (or action) at the leaf level in MCTS satisfy the required concentration property with $\eta = 1/2$ due to independence. Hence, from our result for non-stationary MAB above, we immediately obtain that we can recursively apply the UCB algorithm per (1.2) at each level in the MCTS with $\eta = 1/2$ and appropriately adjusted constants $\beta$ and $\xi$. In effect, we obtain modified UCT where the bias or bonus term $B_{t,s}$ scales as $t^{1/4}/s^{1/2}$. This is in contrast to $B_{t,s}$ scaling as $\sqrt{\log t/s}$ in the standard UCB as well as UCT suggested in the literature [94, 95]. Interestingly enough, the empirical results of AGZ are obtained by utilizing a bonus term that scales as $t^{1/2}/s$. This is qualitatively similar to what our results suggest.

By recursively applying the convergence and concentration property of the non-stationary MAB for the resulting algorithm for MCTS, we establish that for any query state $s$ of the MDP, using $n$ simulations of the MCTS, we can obtain a value function estimation within error $\delta\varepsilon_0 + O\big(n^{-1/2}\big)$, if we start with a value function estimation for all the leaf nodes within error $\varepsilon_0$ for some $\delta < 1$ (independent of $n$, dependent on depth of MCTS tree). That is, MCTS is indeed asymptotically correct as was conjectured in the prior literature.

**MCTS with Supervised Learning (Chapter 2).** The result stated above for MCTS implies its "bootstrapping" property – if we start with a value function estimation for *all* states within error $\varepsilon$, then MCTS can produce estimation of value function for a *given query* state within error less than $\varepsilon$ with enough simulations. By coupling such improved estimations of value function for a number of query states, combined with expressive enough supervised learning, one can hope to generalize such improved estimations of value function for *all* states. That is, MCTS coupled with supervised learning can be a "strong policy improvement operator". Indeed, this is precisely what we establish by utilizing nearest neighbor supervised learning. Specifically, we establish that with $\widetilde{O}\big(\varepsilon^{-(4+d)}\big)$ number of samples, MCTS with nearest neighbor finds an $\varepsilon$ approximation of the optimal value function with respect to $\ell_\infty$-norm; here $d$ is the dimension of the state space. This is nearly optimal in view of a minimax lower bound of $\widetilde{\Omega}\big(\varepsilon^{-(2+d)}\big)$ [143].

**Sample-efficient Low-rank RL (Chapter 3).** Given the universal representation with the notion of dimensionality for $Q^*$ through its rank, we develop a data-efficient RL method.

Specifically, for any $\varepsilon > 0$, our method finds $\hat{Q}$ such that $\|\hat{Q} - Q^*\|_\infty < \varepsilon$ using $\tilde{O}\big(\varepsilon^{-(\max\{d_1,d_2\}+2)}\big)$ samples, with the hidden constant in $\tilde{O}(\cdot)$ dependent on $r, \max\{d_1, d_2\}$. In contrast, the minimax lower bound for learning a generic Lipschitz $Q^*$ in the $L^\infty$ sense (also in the $L^2$-sense) is of $\Omega\big(\varepsilon^{-(d_1+d_2+2)}\big)$ [164]. That is, our method removes the dependence on the smaller of the two dimensions by exploiting the low-rank structure in $Q^*$. Note that this provides an exponential improvement in sample complexity, e.g., with $d_1 = d_2 = d$, our method requires the number of samples scaling as $\varepsilon^{-d-2}$ in contrast to $\varepsilon^{-2d-2}$ required for generic Lipschitz $Q^*$. While low-rank representation of $Q^*$ enables theoretical guarantees, the proof is in the puddling: we find that for well-known control tasks, the underlying $Q^*$ has a low-rank structure. In particular, empirically, using our method that exploits the low-rank structure leads to a significant improvement in sample complexity over the method that does not.

**Matrix Estimation (ME), A Novel Method (Chapter 3).** Our data-efficient RL method relies on a novel low-rank Matrix Estimation method we introduce. Notice that for any set of $m$ states $\{s_k\}_{k=1}^m$ and $n$ actions $\{a_\ell\}_{\ell=1}^n$, the induced matrix $[Q^*(s_k, a_\ell) : k \in [m], \ell \in [n]]$ has rank (at most) $r$. At a high level, to obtain the improved sample complexity as claimed, we wish to faithfully recover the $m \times n$ rank-$r$ matrix in the $\ell_\infty$ sense, by observing only $\tilde{O}\big(\max(m, n)r\big)$ entries with each entry having bounded, but arbitrary noise $\delta$. In the literature [31, 32, 37, 49], such a harsh setting has not been considered. In this work, we introduce a ME method that manages to recover the entire matrix with entry-wise error within $O(\delta)$ through adaptively sampling certain rows and columns. This advance in ME should be of independent interest. With this novel method, we improve our estimates of $Q^*$ iteratively by interleaving one-step lookahead and ME. This, ultimately leads to an $\varepsilon$-optimal $Q^*$ with desired data efficiency.

**Sample Efficient Stable RL Policy (Chapter 4).** As a proof of concept, we present a simple RL policy using a Sparse Sampling Monte Carlo oracle [88] that is stable for any MDP, as long as the optimal policy respects a Lyapunov function with *drift* condition. Our policy *does not* require knowledge of or access to such a Lyapunov function. It recommends an action at each time using finitely many *simulations* of the MDP through the oracle. That is, the policy is *online* and guarantees stability for each trajectory starting *without* any prior training. The number of samples required at each time step scales as $O\Big(\big(\frac{1}{\alpha^4}\log^2\frac{1}{\alpha}\big)^{O(\log\frac{1}{\alpha})}\Big)$, where $-\alpha < 0$ is the drift in Lyapunov function.

To further improve the data efficiency, for MDPs with Lipschitz optimal value function, we propose a modified Sparse Sampling Monte Carlo oracle for which the number of samples required at each time step scales as $O\Big(\frac{1}{\alpha^{2d+4}}\log^{d+1}\frac{1}{\alpha}\Big)$, where $d$ is the dimension of the state space. That is, the sample complexity becomes polynomial in $1/\alpha$ from being super-polynomial with the vanilla oracle. The efficient oracle utilizes the minimal structure of smoothness in the optimal value function and should be of interest in its own right, as it

provides sample complexity improvement for *all* policies in the literature where such an oracle plays a key role, e.g., [89, 128].

**Adaptive Algorithm Based on a Statistical Test (Chapter 4).** While the algorithm does not require knowing the Lyapunov function itself, it does have a parameter whose optimal value depends on the drift parameter of the Lyapunov function. Therefore, we further develop an adaptive, agnostic version of our algorithm that automatically searches for an appropriate tuning parameter. We establish that either this algorithm discovers the right value and hence ensures stability, or the system is near-stable in the sense that $||s_t||/\log^2 t = O(1)$ as $t \to \infty$. The near-stability is a form of sub-linear regret. For example, in the context of a queueing system, this would correspond to queues growing as $O(\log^2 t)$ with time in contrast to $O(1)$ queues for stable (or optimal) policy. Further, in the context of queueing systems, it would imply the so-called "rate" stability [46] — to the best of our knowledge, this is first such general RL policy for generic queueing systems with such a property.

## ■ 1.4 Organization of the Thesis

The rest of the thesis is organized as follows. In Chapter 2, we focus on analyzing the data complexity of MCTS and the resulting learning algorithm when combined with supervised learning. Chapter 3 investigates data-efficient RL algorithms for low-rank $Q^*$ via matrix estimation techniques. In Chapter 4, we study efficient, stable policy for problems with unbounded state space. Each of the chapters is structured to be self-contained with additional necessary background introduced. Finally, we conclude in Chapter 5 with additional remarks and future directions. Appendices A and B provide additional results and discussions for Chapters 2 and 3, respectively.

## ■ 1.5 Bibliographic Note

Preliminary versions of the results in Chapter 2 appeared in [144]. Chapter 3 is based on the work [142]. Finally, earlier versions of the results in Chapter 4 can be found in [145].

Chapter 2

# Non-asymptotic Analysis of Monte Carlo Tree Search

We formally start our investigation on data efficiency of reinforcement learning in this chapter. Motivated by the impressive achievements of AlphaGo Zero (AGZ) [150], we are curious to develop some theoretical understanding about the data efficiency involved in its algorithmic design. To this end, this chapter studies the popular tree-based search strategy within the framework of RL, the Monte Carlo Tree Search (MCTS).

MCTS is a search framework for finding optimal decisions, based on the search tree built by random sampling of the decision space [26]. Since MCTS was first introduced, many variations and enhancements have been proposed, and it has been widely used in sequential decision makings that have a tree structure, exemplified by games and planning problems. In particular, as mentioned in Section 1.2.1, MCTS is arguably one of the key components accounting for the success of AGZ: it is recursively applied to generate training samples, upon which supervised learning methods are employed to learn a policy/value function (represented by deep neural networks).

Despite the wide application and empirical success of MCTS, there is only limited work on theoretical guarantees of MCTS and its variants. Arguably, one of the most important and popular work in this area is [94, 95], which propose running tree search by applying the Upper Confidence Bound (UCB) algorithm — originally designed for stochastic multi-arm bandit (MAB) problems [6, 13] — to each node of the tree for balancing exploration and exploitation within the tree search. This leads to the so-called UCT (Upper Confidence Bounds for Trees) algorithm, which is one of the popular forms of MCTS in practice. While UCT is believed to provide an approximately optimal value function for a given state with enough simulations [94, 95], the claimed proof of this property is incomplete, as we discuss in greater details in Section 2.1. More importantly, naively following the insights from the stochastic MAB literature is problematic. With logarithmic bonus in [94], UCT as suggested in effect requires exponential concentration of regret for the underlying non-stationary MAB associated with each node in the tree, which is unlikely to hold in general even for stationary

MAB as pointed out in [12]. .

Indeed, rigorous analysis of MCTS is subtle, even though its asymptotic convergence may seem natural. A key challenge is that the tree policy (e.g., UCT) for selecting actions typically needs to balance exploration and exploitation, so the random sampling process at each node is non-stationary (non-uniform) across multiple simulations. A more severe difficulty arises due to the hierarchical/iterative structure of tree search, which induces complicated probabilistic dependency between a node and the nodes within its sub-tree. Specifically, as part of simulation within MCTS, at each intermediate node (or state), the action is chosen based on the outcomes of the past simulation steps within the sub-tree of the node in consideration. Such strong dependencies across time (i.e., depending on the history) and space (i.e., depending on the sub-trees downstream) amongst nodes make the analysis non-trivial.

As the key contribution of this chapter, we establish polynomial concentration property of regret for a class of *non-stationary* MAB. We show that the non-stationary reward process at each node of the simulation tree indeed respects such a polynomial concentration. This in turn establishes that the MCTS with appropriate *polynomial* rather than *logarithmic* bonus term in UCB has the claimed property of [94, 95]. Interestingly enough, empirically successful approaches [150] utilize a similar polynomial form of MCTS as suggested by our result (cf. Section 2.3.2). Using this as a building block, we analyze a stylized, abstract version of AGZ's algorithm by combining MCTS with supervised learning iteratively. In particular, we argue that MCTS, combined with nearest neighbor supervised learning, acts as a "policy improvement" operator, i.e., it iteratively improves value function approximation for *all* states, due to combining with supervised learning, despite evaluating at only finitely many states. In effect, we establish that to learn an $\varepsilon$-optimal value function with respect to $\ell_\infty$ norm, MCTS combined with nearest neighbor requires a sample size scaling as $\widetilde{O}\big(\varepsilon^{-(d+4)}\big)$, where $d$ is the dimension of the state space. This is nearly optimal due to a minimax lower bound of $\widetilde{\Omega}\big(\varepsilon^{-(d+2)}\big)$, suggesting the strength of the variant of MCTS we propose here and our resulting analysis.

**Organization of Chapter 2.** We structure Chapter 2 as follows. To begin with, a literature survey on MCTS is provided in Section 2.1. Then, Section 2.2 states the formal setting considered and the conditions required in this chapter. With this background, Section 2.3 describes the Monte Carlo Tree Search algorithm and our main result on its non-asymptotic analysis. Section 2.4 then focuses on a reinforcement learning method that combines MCTS with nearest neighbor supervised learning. It describes the finite-sample guarantees of the method for finding $\varepsilon$-optimal value function with respect to $\ell_\infty$ norm. The remaining sections of this chapter provide all the technical details. In order to prove our claims, Section 2.5 introduces a form of non-stationary multi-arm bandit and an upper confidence bound policy

for it. For this setting, we present the concentration of induced regret and prove it in Section 2.6. This is a key result and serves as the building block, with which we establish the desired property of MCTS in Section 2.7. Section 2.8 finishes the proof for the guarantees of the RL method using MCTS. Further, with the main insights developed, we extend our results to the general stochastic setting in Section 2.9. Finally, we conclude this chapter in Section 2.10. Supplementary results on the proof of a related lower bound as well as toy numerical experiments are provided in Appendices A.1 and A.2, respectively.

## ■ 2.1 Related Work

We discuss some of the existing work on MCTS and its various modifications used for RL. As previewed, MCTS is an approach for estimating the (optimal) value of states by building a search tree from Monte-Carlo simulations [94, 33, 45, 26]. In [94, 95], authors argue for the asymptotic convergence of MCTS with standard UCT. However, the proof is incomplete [160]. A key step towards proving the claimed result is to show the convergence and concentration properties of the regret for UCB under non-stationary reward distributions. In particular, to establish an exponential concentration of regret (Theorem 5, [95]), Lemma 14 is applied. However, it requires conditional independence of $\{Z_i\}$ sequence, which does not hold, hence making the conclusion of exponential concentration questionable. Therefore, the proof of the main result (Theorem 7, [95]), which applies Theorem 5 with an inductive argument, is a conjecture at best.

In fact, it may be infeasible to prove Theorem 5 in [95] as it was stated. For example, the work of [12] shows that for bandit problems, the regret under UCB concentrates around its expectation polynomially, rather than *exponentially* as desired in [95] (e.g., if the essential infimum of the optimal arm's reward is below the mean reward of the second-best arm; see Theorem 10 of [12]). Further, authors in [137] prove that for any strategy that does not use the knowledge of time horizon, it is infeasible to improve this polynomial concentration and establish exponential concentration. Our result is consistent with these fundamental bounds of stationary MAB — we establish polynomial concentration of regret for non-stationary MAB, which plays a crucial role in our analysis of MCTS. Also see the work [124] for a discussion of the issues with logarithmic bonus terms for tree search.

While we focus on UCT in this chapter, we note that there are other variants of MCTS developed for a diverse range of applications. The work of [44] introduces flat UCB in order to improve the worst case regret bounds of UCT. In [138], MCTS is modified for single-player games by adding to the standard UCB formula a term that captures the possible deviation of the node. In the work by [156], a variant of MCTS is introduced for multi-player games by adopting the $\max^n$ idea. In addition to turn-based games like Go and Chess, MCTS has also

been applied to real-time games (e.g., Ms. PacMan, Tron and Starcraft) and nondeterministic games with imperfect information. The applications of MCTS go beyond games, and appear in areas such as optimization, scheduling and other decision-making problems. We refer to the survey on MCTS by [26] for other variations and applications.

It has become popular recently to combine MCTS with deep neural networks, which serve to approximate the value function and/or policy [148, 150, 149]. For instance, in AlphaGo Zero, MCTS uses the neural network to query the value of leaf nodes for simulation guidance; the neural network is then updated with sample data generated by MCTS-based policy and used in tree search in the next iteration. The work of [16] develops generative adversarial tree search that generates roll-outs with a learned GAN-based dynamic model and reward predictor, while using MCTS for planning over the simulated samples and a deep $Q$-network to query the $Q$-value of leaf nodes.

In terms of theoretical results, the closest work to this chapter is [79], where they also consider a batch, MCTS-based reinforcement learning algorithm, which is a variant of AlphaGo Zero's algorithm. The key algorithmic difference from ours lies in the leaf-node evaluator of the search tree: they use a combination of an estimated value function and an estimated policy. The latest observations at the root node are then used to update the value and policy functions (leaf-node evaluator) for the next iteration. They also give a finite-sample analysis. However, their result and ours are quite different: in their analysis, the sample complexity of MCTS, as well as the approximation power of value/policy architectures, are *imposed as an assumption*; here we *prove* an explicit finite-sample bound for MCTS and characterize the non-asymptotic error prorogation under MCTS with non-parametric regression for leaf-node evaluation. Therefore, they *do not* establish "strong policy improvement" property of the MCTS.

Two other closely related papers are [162] and [87], which study a simplified MCTS for two-player zero-sum games. There, the goal is to identify the best action of the root in a *given* game tree. For each leaf node, a stochastic oracle is provided to generate i.i.d. samples for the true reward. In [162], authors give a high probability bound on the number of oracle calls needed for obtaining $\varepsilon$-accurate score at the root. The more recent paper [87] develops refined, instance-dependent sample complexity bounds. Compared to classical MCTS (e.g., UCT), both the setting and the algorithms in the above papers are simpler: the game tree is given in advance, rather than being built gradually through samples; the algorithm proposed in [162] operates on the tree in a bottom-up fashion with uniform sampling at the leaf nodes. As a result, the analysis is significantly simpler and it is unclear whether the techniques can be extended to analyze other variants of MCTS.

It is important to mention the work of [33] that explores the idea of using UCB for adaptive sampling in MDPs. The approximate value computed by the algorithm is shown

to converge to the optimal value. We remark that their algorithm is different from the algorithm we analyze in this chapter. In particular, their algorithm proceeds in a depth-first, recursive manner, and hence involves using UCB for a stationary MAB at each node. In contrast, the UCT algorithm we study involves non-stationary MABs, hence our analysis is significantly different from theirs. We refer the readers to the work by [94] and [45] for further discussion of the difference. Another related work by [88] studies a sparse sampling algorithm for large MDPs. This algorithm is also different from the MCTS family we analyze in this chapter. Relatedly, [14] considers a setting with finite horizon and continuous action space. During the tree simulation, a progressive widening technique is used to decide when to sample (add) a new action at each step; if no new action is needed for the current step, UCT is then extended with a *specific* choice and parameter of polynomial bonus for action selection. In contrast, we consider an infinite horizon setting. More importantly, we establish guarantees for a *class* of polynomial bonus forms determined by the set of inter-dependent algorithmic parameters. Again, this is made possible by introducing an appropriate form of non-stationary MAB, which could be of independent interest. Recently, this idea is further extended by [117] to establish results for MCTS with a continuous armed bandit strategy, which shows more favorable performance than the algorithm proposed by [14]. Finally, we remark that the work by [56] studies multiple-step lookahead policies in RL, which can be implemented via MCTS.

## ■ 2.2 Setup and Problem Statement

### ■ 2.2.1 MDP Regularity

We consider the setup of infinite-horizon discounted Markov Decision Process (MDP) as reviewed in Section 1.1. To recap, this is described by a five-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{P} \equiv \mathcal{P}(s'|s,a)$ is the Markovian transition kernel, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a random reward function, and $\gamma \in (0,1)$ is a discount factor. At each time step, the system is in some state $s \in \mathcal{S}$. When an action $a \in \mathcal{A}$ is taken, the state transits to a next state $s' \in \mathcal{S}$ according to the transition kernel $\mathcal{P}$ and an immediate reward is generated as an independent sample according to the distribution of $\mathcal{R}(s,a)$. Again, We consider the setup with access to the generative model (i.e., a simulator) [84], which is a common setting in the theoretical reinforcement learning literature. Precisely, we assume that the agent has knowledge of $\mathcal{S}$, $\mathcal{A}$ and $\gamma$. The transition kernel $\mathcal{P}$ and the rewards $\mathcal{R}$ are unknown, but the agent could query the generative model at any given state-action pair $(s,a)$ to obtain a sample of next state and the associated immediate reward.

In this chapter, we focus on estimating the optimal value function $V^*$. Recall that the value function $V^\pi(s)$ associated with a policy $\pi$ is defined as the expected discounted sum

of rewards received by following the policy $\pi$ from the initial state $s$, and $V^*$ is the value function of the reward-maximizing policy, i.e., $V^*(s) = V^{\pi^*}(s) = \sup_\pi V^\pi(s)$, $\forall s \in \mathcal{S}$. It is well understood that such an optimal policy/value exists in reasonable generality [23]. In this chapter, we restrict our attention to the MDPs with the following regularity conditions.

1. (Compact domain) The action space $\mathcal{A}$ is a finite set and the state space $\mathcal{S}$ is a compact subset of $d$ dimensional set; without loss of generality, let $\mathcal{S} = [0,1]^d$;

2. (Bounded reward) The immediate rewards are random variables, uniformly bounded such that $\mathcal{R}(s,a) \in [-R_{\max}, R_{\max}]$, $\forall s \in \mathcal{S}, a \in \mathcal{A}$ for some $R_{\max} > 0$;

3. (Deterministic transition) The state transitions are deterministic, i.e. $\mathcal{P} \equiv \mathcal{P}(s'|s,a) \in \{0,1\}$ for all $s, s' \in \mathcal{S}$, $a \in \mathcal{A}$.

Define $\beta \triangleq 1/(1-\gamma)$ and $V_{\max} \triangleq \beta R_{\max}$. Since all the rewards are bounded by $R_{\max}$, it is easy to see that the absolute value of the value function for any state under any policy is bounded by $V_{\max}$ [58, 155].

**On Deterministic Transition.** Traditional AI game research has been focused on deterministic games with a tree representation. MCTS has been extensively utilized in such deterministic transition problems [26], as demonstrated by the recent successes of MCTS in Go [150], Chess [149] and Atari games [67]. There has been an extensive theoretical literature on the analysis of MCTS and related methods for deterministic transitions [26, 77, 124, 17], which provide crucial insights for more general scenarios in RL.

Having noted that, our analysis and results for deterministic transitions indeed naturally extend to the stochastic setting with minor modifications. Considering the importance of deterministic transition setting, as well as the clarity of our proof framework, we first develop the results and the associated analysis for the setting of deterministic transitions. After developing a good understanding of the main ideas, we shall extend them for the stochastic setting as described in Section 2.9.

### ■ 2.2.2 Value Function Iteration

We review the classical value function iteration which will be useful for our technical development. It is an iterative approach for finding optimal value function, $V^*$. To begin, the so-called Bellman equation characterizes the optimal value function as

$$V^*(s) = \max_{a \in \mathcal{A}} \Big( \mathbb{E}[\mathcal{R}(s,a)] + \gamma V^*(s \circ a) \Big), \tag{2.1}$$

where $s \circ a \in \mathcal{S}$ is the notation to denote the state reached by applying action $a$ on state $s$. Under our considered setup, the transitions are deterministic and hence $s \circ a$ represents a

single, deterministic state rather than a random state.

The value function iteration effectively views (2.1) as a fixed point equation and tries to find a solution to it through a natural iteration. Precisely, let $V^{(t)}(\cdot)$ be the value function estimation in iteration $t$ with $V^{(0)}$ being arbitrarily initialized. Then, for $t \geq 0$, for all $s \in \mathcal{S}$,

$$V^{(t+1)}(s) = \max_{a \in \mathcal{A}} \Big( \mathbb{E}[\mathcal{R}(s,a)] + \gamma V^{(t)}(s \circ a) \Big). \tag{2.2}$$

It is well known [23] that value iteration is contractive with respect to $\|\cdot\|_\infty$ norm for all $\gamma < 1$. Specifically, for $t \geq 0$, we have

$$\|V^{(t+1)} - V^*\|_\infty \leq \gamma \|V^{(t)} - V^*\|_\infty. \tag{2.3}$$

## ■ 2.3 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) has been quite popular recently in many of the reinforcement learning tasks. In effect, given a state $s \in \mathcal{S}$ and a value function estimate $\hat{V}$, it attempts to run the value function iteration for a fixed number of steps, say $H$, to evaluate $V^{(H)}(s)$ starting with $V^{(0)} = \hat{V}$ per (2.2). This, according to (2.3), would provide an estimate within error $\gamma^H \|\hat{V} - V^*\|_\infty$ – an excellent estimate of $V^*(s)$ if $H$ is large enough. The goal is to perform computation for value function iteration necessary to evaluate $V^{(H)}$ for state $s$ only and not necessarily for all states as required by traditional value function iteration. MCTS achieves this by simply "unrolling" the associated "computation tree". Another challenge that MCTS overcomes is the fact that value function iteration as in (2.2) assumes knowledge of model so that it can compute $\mathbb{E}[\mathcal{R}(\cdot,\cdot)]$ for any state-action pair. But in reality, rewards are observed through samples, not a direct access to $\mathbb{E}[\mathcal{R}(\cdot,\cdot)]$. MCTS tries to utilize the samples in a careful manner to obtain accurate estimation for $V^{(H)}(s)$ over the computation tree suggested by the value function iteration as discussed above. The concern of careful use of samples naturally connects it to multi-arm bandit like setting. We remark that in this chapter, our goal is to understand the theoretical (finite-sample) property of MCTS in estimating the value of the query state. Practical usage of MCTS may also involve constructing and executing a policy for the query state based on the simulation data.

Next, we present a detailed description of the MCTS algorithm in Section 2.3.1. This can be viewed as a *correction* of the algorithm presented in [94, 95]. We state its theoretical property in Section 2.3.2.

### ■ 2.3.1 Algorithm

We provide details of a specific form of MCTS, which replaces the logarithmic bonus

**Algorithm 1** Fixed-Depth Monte Carlo Tree Search

**Input:** (1) current value oracle $\hat{V}$, root node $s^{(0)}$ and search depth $H$;
(2) number of MCTS simulations $n$;
(3) algorithmic constants, $\{\alpha^{(i)}\}_{i=1}^{H}$, $\{\beta^{(i)}\}_{i=1}^{H}$, $\{\xi^{(i)}\}_{i=1}^{H}$ and $\{\eta^{(i)}\}_{i=1}^{H}$.

**Initialization:** for each depth $h$, initialize the cumulative node value $\tilde{v}^{(h)}(s) = 0$ and visit count $N^{(h)}(s) = 0$ for every node $s$ and initialize the cumulative edge value $q^{(h)}(s,a) = 0$.

**for** each MCTS simulation $t = 1, 2, \ldots, n$ **do**

`/* Simulation: select actions until reaching depth` $H$`*/`

**for** depth $h = 0, 1, 2, \ldots, H-1$ **do**

at state $s^{(h)}$ of depth $h$, select an action (edge) according to

$$a^{(h+1)} = \arg\max_{a \in \mathcal{A}} \frac{q^{(h+1)}(s^{(h)}, a) + \gamma \tilde{v}^{(h+1)}(s^{(h)} \circ a)}{N^{(h+1)}(s^{(h)} \circ a)} + \frac{\left(\beta^{(h+1)}\right)^{1/\xi^{(h+1)}} \cdot \left(N^{(h)}(s^{(h)})\right)^{\alpha^{(h+1)}/\xi^{(h+1)}}}{\left(N^{(h+1)}(s^{(h)} \circ a)\right)^{1-\eta^{(h+1)}}}, \tag{2.4}$$

where dividing by zero is assumed to be $+\infty$.

upon taking the action $a^{(h+1)}$, receive a random reward $r^{(h+1)} \triangleq \mathcal{R}(s^{(h)}, a^{(h+1)})$ and transit to a new state $s^{(h+1)}$ at depth $h+1$.

**end for**

`/* Evaluation: call value oracle for leaf nodes*/`

reach $s^{(H)}$ at depth $H$, call the current value oracle and let $\tilde{v}^{(H)}(s^{(H)}) = \hat{V}(s^{(H)})$.

`/* Update Statistics: quantities on the search path*/`

**for** depth $h = 0, 1, 2, \ldots, H-1$ **do**

update statistics of nodes and edges that are on the search path of current simulation:

visit count: $N^{(h+1)}(s^{(h+1)}) = N^{(h+1)}(s^{(h+1)}) + 1$

edge value: $q^{(h+1)}(s^{(h)}, a^{(h+1)}) = q^{(h+1)}(s^{(h)}, a^{(h+1)}) + r^{(h+1)}$

node value: $\tilde{v}^{(h)}(s^{(h)}) = \tilde{v}^{(h)}(s^{(h)}) + r^{(h+1)} + \gamma r^{(h+2)} + \cdots + \gamma^{H-1-h} r^{(H)} + \gamma^{H-h} \tilde{v}^{(H)}(s^{(H)})$

**end for**

**end for**

**Output:** average of the value for the root node $\tilde{v}^{(0)}(s^{(0)})/n$.

term of UCT with a polynomial one. Overall, we fix the search tree to be of depth $H$. Similar to most literature on this topic, it uses a variant of the Upper Confidence Bound (UCB) algorithm to select an action at each stage (depth) in order to balance exploration and exploitation. That is, at a node, the action that corresponds to the highest mean reward plus an appropriate bonus term thus far is selected. A state at the next depth is then reached and the action selection process is repeated. At a leaf node (i.e., a state at depth $H$), we use the current value oracle $\hat{V}$ to evaluate its value, and this finishes one iteration of MCTS. Hence, one iteration of MCTS corresponds to one length $H$ path from

the root to the leaf. The rewards collected along the simulated path are then utilized to update values corresponding to the nodes/edges for simulation guidance at the subsequent iteration. Note that since we consider deterministic transitions, consequently, the tree is fixed once the root node (state) is chosen, and we use the notation $s \circ a$ to denote the next state after taking action $a$ at state $s$. Each edge represents a state-action pair, while each node represents a state. For clarity, we use superscript to distinguish quantities related to different depth. The pseudo-code for the MCTS procedure is given in Algorithm 1, and Figure 2.1 shows the structure of the search tree and related notation.

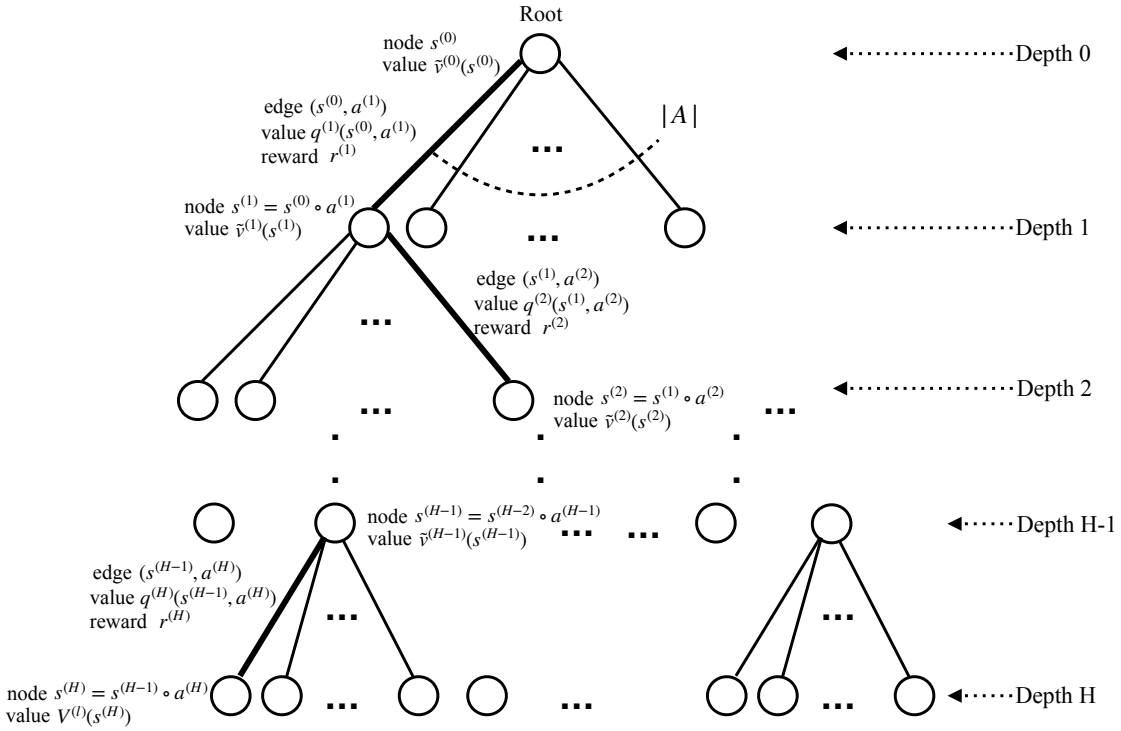



**Figure 2.1.** Notation and a sample simulation path of MCTS (thick lines).

In Algorithm 1, there are certain sequences of algorithmic parameters required, namely, $\alpha$, $\beta$, $\xi$ and $\eta$. The choices for these constants will become clear in our non-asymptotic analysis. At a higher level, the constants for the last layer (i.e., depth $H$), $\alpha^{(H)}$, $\beta^{(H)}$, $\xi^{(H)}$ and $\eta^{(H)}$ depend on the properties of the leaf nodes, while the rest are recursively determined by the constants one layer below. We note that in selecting action $a^{(h+1)}$ at each depth $h$ (i.e., Line 6 of Algorithm 1), the upper confidence term is polynomial in $n$ while a typical UCB algorithm would be logarithmic in $n$, where $n$ is the number of visits to the corresponding state thus far. The logarithmic factor in the original UCB algorithm was motivated by the exponential tail probability bounds. In our case, it turns out that exponential tail bounds for each layer seems to be infeasible without further structural assumptions. As mentioned

in Section 2.1, prior work [12, 137] has justified the polynomial concentration of the regret for the classical UCB in stochastic (independent rewards) multi-arm bandit setting. This implies that the concentration at intermediate depth (i.e., depth less than $H$) is at most polynomial. Indeed, we will prove these polynomial concentration bounds even for non-stationary (dependent, non-stationary rewards) multi-arm bandit that shows up in MCTS and discuss separately in Section 2.5.

### ■ 2.3.2 Analysis

Now, we state the following result on the non-asymptotic performance of the MCTS as described above.

**Theorem 1.** *Consider an MDP satisfying the regularity conditions in Section 2.2.1. Let $H \geq 1$, and for $1/2 \leq \eta < 1$, let*

$$\eta^{(h)} = \eta^{(H)} \equiv \eta, \quad \forall\, h \in [H], \tag{2.5}$$

$$\alpha^{(h)} = \eta(1-\eta)\big(\alpha^{(h+1)} - 1\big), \quad \forall\, h \in [H-1], \tag{2.6}$$

$$\xi^{(h)} = \alpha^{(h+1)} - 1, \quad \forall\, h \in [H-1]. \tag{2.7}$$

*Suppose that a large enough $\xi^{(H)}$ is chosen such that $\alpha^{(1)} > 2$. Then, there exist corresponding constants $\{\beta^{(i)}\}_{i=1}^{H}$ such that for each query state $s \in \mathcal{S}$, the following claim holds for the output $\hat{V}_n(s)$ of MCTS with $n$ simulations:*

$$\Big|\mathbb{E}\big[\hat{V}_n(s)\big] - V^*(s)\Big| \leq \gamma^H \varepsilon_0 + O\Big(n^{\eta-1}\Big), \tag{2.8}$$

*where $\varepsilon_0 = \|\hat{V} - V^*\|_\infty$ with $\hat{V}$ being the estimate of $V^*$ utilized by the MCTS algorithm for leaf nodes.*

Since $\eta \in [1/2, 1)$, Theorem 1 implies a best case convergence rate of $O(n^{-1/2})$ by setting $\eta = 1/2$. We note that the constant in the $O(\cdot)$-notation also depends on $\eta \in [\frac{1}{2}, 1)$. However, the leading impact of $\eta$ on $n$ is entirely captured through $n^{\eta-1}$. Therefore, the order-wise optimal convergence is achieved by the choice of $\eta = 1/2$. With these parameter choices, the bonus term in the upper confidence bound (line 6 of Algorithm 1) scales as $\big(N^{(h)}(s^{(h)})\big)^{1/4}/\sqrt{N^{(h+1)}(s^{(h)} \circ a)}$, that is, in the form of $t^{1/4}/\sqrt{S}$, where $t \equiv N^{(h)}(s^{(h)})$ is the number of times that state $s^{(h)}$ at depth $h$ has been visited, and $S \equiv N^{(h+1)}(s^{(h)} \circ a)$ is the number of times action $a$ has been selected at state $s^{(h)}$. Interestingly enough, the empirical results of AGZ are obtained by utilizing a bonus that scales as $t^{1/2}/S$. This is qualitatively similar to what our results suggest.

**High Probability Bound.** Theorem 1 states bounds on expected estimation error in value

function, cf. (2.8). We remark that the proof is established via recursively arguing a certain form of convergence and polynomial concentration properties for the non-stationary value function estimate sequence for nodes at each depth. That is, starting with the convergence and polynomial concentration properties for nodes at depth $h+1$, we establish a similar form of convergence and polynomial concentration properties for nodes at depth $h$. We recursively apply this argument, starting from the leaf nodes, until reaching the root node. Therefore, the output $\hat{V}_n(s)$ of MCTS at root node also satisfies a form of polynomial concentration. Specifically, under the setup of Theorem 1, it follows that for every $n \geq 1$ and every $z \geq 1$

$$\mathbb{P}\big(n\hat{V}_n(s) - n\mu^*(s) \geq n^\eta z\big) \leq \frac{\beta^{(1)}}{z^{\xi^{(0)}}}, \quad \mathbb{P}\big(n\hat{V}_n(s) - n\mu^*(s) \leq -n^\eta z\big) \leq \frac{\beta^{(1)}}{z^{\xi^{(0)}}},$$

where $\eta$, $\xi^{(0)}$ and $\beta^{(1)}$ are some constants (see Theorem 1 and the proof in Section 2.7 for details). Here, $\mu^*(s)$ is the value function estimation for $s$ after $H$ iterations of value function iteration starting with $\hat{V}$. With the classical contraction result for value function iteration, i.e., $|\mu^*(s) - V^*(s)| \leq \gamma^H \varepsilon_0$, we obtain

$$\mathbb{P}\big(n\hat{V}_n(s) - nV^*(s) \geq n^\eta z + \gamma^H \varepsilon_0\big) \leq \frac{\beta^{(1)}}{z^{\xi^{(0)}}}, \quad \mathbb{P}\big(n\hat{V}_n(s) - nV^*(s) \leq -n^\eta z - \gamma^H \varepsilon_0\big) \leq \frac{\beta^{(1)}}{z^{\xi^{(0)}}}.$$

## ■ 2.4 Reinforcement Learning through MCTS with Supervised Learning

Recently, MCTS has been utilized prominently in various empirical successes of reinforcement learning including AlphaGo Zero (AGZ). Here, MCTS is combined with expressive supervised learning methods to iteratively improve the policy as well as the value function estimation. In effect, MCTS combined with supervised learning acts as a "policy improvement" operator.

Intuitively, MCTS produces an improved estimation of value function for a given state of interest, starting with a given estimation of value function by "unrolling" the "computation tree" associated with value function iteration. And MCTS achieves this using observations obtained through simulations. Establishing this improvement property rigorously was the primary goal of Section 2.3. Now, given such improved estimation of value function for finitely many states, a good supervised learning method can learn to generalize such an improvement to all states. If so, this is like performing value function iteration, but using simulations. Presenting such a policy and establishing such guarantees is the crux of this section.

To that end, we present a reinforcement learning method that combines MCTS with nearest neighbor supervised learning. For this method, we establish that indeed, with suf-

ficient number of samples, the resulting policy improves the value function estimation just like value function iteration. Using this, we provide a finite-sample analysis for learning the optimal value function within a given tolerance. We find it nearly matching a minimax lower bound in [143] which we recall in Section 2.4.4, and thus establish near minimax optimality of such a reinforcement learning method.

### ■ 2.4.1 Reinforcement Learning Procedure

Here we describe the method to produce estimation of optimal value function $V^*$. Similar approach can be applied to obtain estimation of policy as well. Let $V^{(0)}$ be the initial estimation of $V^*$, and for simplicity, let $V^{(0)}(\cdot) = 0$. We describe a procedure that iterates between use of MCTS and supervised learning to iteratively obtain estimation $V^{(\ell)}$ for $\ell \geq 1$, so that iteratively better estimation of $V^*$ is produced as $\ell$ increases. To that end, for $\ell \geq 1$,

- For appropriately sampled states $S^\ell = \{s_i\}_{i=1}^{m_\ell}$, apply MCTS to obtain improved estimations of value function $\{\hat{V}^{(\ell)}(s_i)\}_{i=1}^{m_\ell}$ using $V^{(\ell-1)}$ to evaluate leaf nodes during simulations.
- Using $\{(s_i, \hat{V}^{(\ell)}(s_i)\}_{i=1}^{m_\ell}$ with a variant of nearest neighbor supervised learning with parameter $\delta_\ell \in (0, 1)$, produce estimation $V^{(\ell)}$ of the optimal value function.

For completeness, the pseudo-code is provided in Algorithm 2.

**Algorithm 2** Reinforcement Learning Procedure

**Input:** initial value function oracle $V^{(0)}(s) = 0$, $\forall\, s \in \mathcal{S}$
**for** $l = 1, 2, \ldots, L$ **do**
  `/* improvement via MCTS */`
  uniformly and independently sample states $S^\ell = \{s_i\}_{i=1}^{m_\ell}$.
  **for** each sampled state $s_i$ **do**
    apply the MCTS algorithm, which takes as inputs the current value oracle $V^{(l-1)}$, the depth $H^{(l)}$, the number of simulation $n_l$, and the root node $s_i$, and outputs an improved estimate for $V^*(s_i)$:

$$\hat{V}^{(l)}(s_i) = \mathrm{MCTS}\big(V^{(l-1)}, H^{(l)}, n_l, s_i\big) \tag{2.9}$$

  **end for**
  `/* supervised learning */`
  with the collected data $\mathcal{D}^{(l)} = \{(s_i, \hat{V}^{(l)}(s_i))\}_{i=1}^{m_l}$, build a new value oracle $V^{(l)}$ via a nearest neighbor regression with parameter $\delta_l$ :

$$V^{(l)}(s) = \text{Nearest Neigbhor}\big(\mathcal{D}^{(l)}, \delta_l, s\big),\ \forall\, s \in \mathcal{S}. \tag{2.10}$$

**end for**
**Output:** final value oracle $V^{(L)}$.

### ■ 2.4.2 Supervised Learning

For simplicity, we shall utilize the following variant of the nearest neighbor supervised learning parametrized by $\delta \in (0,1)$. Given state space $\mathcal{S} = [0,1]^d$, we wish to cover it with minimal (up to scaling) number of balls of radius $\delta$ (with respect to $\ell_2$-norm). To that end, since $\mathcal{S} = [0,1]^d$, one such construction is where we have balls of radius $\delta$ with centers being $\{(\theta_1, \theta_2, \ldots, \theta_d) : \theta_1, \ldots, \theta_d \in \mathcal{Q}(\delta)\}$ where

$$\mathcal{Q}(\delta) = \left\{\frac{1}{2}\delta i \ : \ i \in \mathbb{Z}, 0 \le i \le \left\lfloor \frac{2}{\delta} \right\rfloor\right\} \cup \left\{1 - \frac{1}{2}\delta i \ : \ i \in \mathbb{Z}, 0 \le i \le \left\lfloor \frac{2}{\delta} \right\rfloor\right\}.$$

Let the collection of these balls be denoted by $c_1, \ldots, c_{K(\delta,d)}$ with $K(\delta, d) = |\mathcal{Q}(\delta)|$. It is easy to verify that $\mathcal{S} \subset \cup_{i \in [K(\delta,d)]} c_i$, $K(\delta, d) = \Theta(\delta^{-d})$ and $C_d \delta^d \le \mathsf{volume}(c_i \cap \mathcal{S}) \le C'_d \delta^d$ for strictly positive constants $C_d, C'_d$ that depends on $d$ but not $\delta$. For any $s \in \mathcal{S}$, let $j(s) = \min\{j : s \in c_j\}$. Given observations $\{(s_i, \hat{V}^{(\ell)}(s_i)\}_{i=1}^{m_\ell}$, we produce an estimate $V^{(\ell)}(s)$ for all $s \in \mathcal{S}$ as follows:

$$V^{(\ell)}(s) = \begin{cases} \frac{\sum_{i: s_i \in c_{j(s)}} \hat{V}^{(\ell)}(s_i)}{|\{i : s_i \in c_{j(s)}\}|}, & \text{if } \ |\{i : s_i \in c_{j(s)}\}| \neq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.11}$$

It is worth noting that other supervised learning algorithms could be used to achieve similar performance guarantees under appropriate conditions. In this work, as a concrete example, we instantiate the supervised learning algorithm with nearest neighbors for its simplicity as well as its generalization guarantee for smooth functions. As only the basic Lipschitz smoothness will be assumed (Assumption 1 in Section 2.4.3), we do not expect order-wise gain in terms of improving sample complexity from other learning methods. However, it could be beneficial to use a more refined method, e.g., local polynomial interpolation, if the underlying function posses higher-order smoothness or a parametric form.

### ■ 2.4.3 Finite-sample Analysis

For finite-sample analysis of the proposed reinforcement learning method, we make the following structural assumption about the MDP. Specifically, we assume that the optimal value function (i.e., true regression function) is smooth in some sense. We note that some form of smoothness assumption for MDPs with continuous state/action space is typical for $\ell_\infty$ guarantee. The Lipschitz continuous assumption stated below is natural and representative in the literature on MDPs with continuous state spaces [124, 55, 54, 21].

**Assumption 1** (Smoothness)**.** *The optimal value function $V^* : \mathcal{S} \to \mathbb{R}$ satisfies Lipschitz continuity with parameter $\zeta$, i.e., $\forall s, s' \in \mathcal{S} = [0,1]^d$, $|V^*(s) - V^*(s')| \le \zeta \|s - s'\|_2$.*

Now we state the result characterizing the performance of the reinforcement learning policy described above. Details of the statement and the proof can be found in Section 2.8.

**Theorem 2.** *Consider a MDP satisfying the regularity conditions in Section 2.2.1 and Assumption 1. Let $\varepsilon > 0$ be a given error tolerance. Then, for $L = \Theta\Big(\log \frac{\varepsilon}{V_{\max}}\Big)$, with appropriately chosen $m_\ell, \delta_\ell$ for $\ell \in [L]$ as well as parameters in MCTS, the reinforcement learning algorithm produces estimation of value function $V^{(L)}$ such that*

$$\mathbb{E}\big[\sup_{s\in\mathcal{S}} |V^{(L)}(s) - V^*(s)|\big] \le \varepsilon,$$

*by selecting $m_\ell$ states uniformly at random in $\mathcal{S}$ within iteration $\ell$. This, in total, requires $T$ number of state transitions (or samples), where*

$$T = O\Big(\varepsilon^{-\big(4+d\big)} \cdot \big(\log \frac{1}{\varepsilon}\big)^5\Big).$$

We note that in Theorem 2, the expectation is taken with respect to all the randomness in the algorithm, i.e., the randomness in sampling the states at each iteration (Line 4 of Algorithm 2) and the randomness in each query of the MCTS algorithm (Line 6 of Algorithm 2).

### ■ 2.4.4 Minimax Lower Bound

Leveraging the minimax lower bound for the problem of non-parametric regression [164, 154], recent work [143] establishes a lower bound on the sample complexity for reinforcement learning algorithms for general MDPs without additional structural assumptions. Indeed, the lower bound also holds for MDPs with deterministic transitions (the proof is provided in Appendix A.1), which is stated in the following proposition.

**Proposition 1.** *Given an algorithm, let $V_T$ be the estimation of $V^*$ after $T$ samples of state transitions for the given MDP. Then, for each $\varepsilon \in (0, 1)$, there exists an instance of deterministic MDP such that in order to achieve*

$$\mathbb{P}\Big(\big\|\hat{V}_T - V^*\big\|_\infty < \varepsilon\Big) \ge 1 - \varepsilon,$$

*it must be that*

$$T \ge C' d \cdot \varepsilon^{-(d+2)} \cdot \log(\frac{1}{\varepsilon}),$$

*where $C' > 0$ is a constant independent of the algorithm.*

Proposition 1 states that for any procedure to learn the optimal value function within $\varepsilon$ approximation error, the number of samples required must scale as $\widetilde{\Omega}\big(\varepsilon^{-(2+d)}\big)$. Theorem 2

implies that the sample complexity of the proposed algorithm scales as $\widetilde{O}\big(\varepsilon^{-(4+d)}\big)$ (omitting the logarithmic factor). Hence, in terms of the dependence on the dimension, the proposed algorithm is nearly optimal. Optimizing the dependence of the sample complexity on other parameters is an important direction for future work.

## ■ 2.5 Non-stationary Multi-arm Bandit

In the next sections, we provide the proofs of the results stated before. We begin by introducing a class of non-stationary multi-arm bandit (MAB) problems, which will play a crucial role in analyzing the MCTS algorithm. To that end, let there be $K \geq 1$ arms or actions of interest. Let $X_{i,t}$ denote the random reward obtained by playing the arm $i \in [K]$ for the $t$th time with $t \geq 1$. Let $\bar{X}_{i,n} = \frac{1}{n}\sum_{t=1}^{n} X_{i,t}$ denote the empirical average of playing arm $i$ for $n$ times, and let $\mu_{i,n} = \mathbb{E}[\bar{X}_{i,n}]$ be its expectation. For each arm $i \in [K]$, the reward $X_{i,t}$ is bounded in $[-R, R]$ for some $R > 0$, and we assume that the reward sequence, $\{X_{i,t} : t \geq 1\}$, is a non-stationary process satisfying the following convergence and concentration properties:

A. (Convergence) the expectation $\mu_{i,n}$ converges to a value $\mu_i$, i.e.,

$$\mu_i = \lim_{n\to\infty} \mathbb{E}[\bar{X}_{i,n}]. \tag{2.12}$$

B. (Concentration) there exist three constants, $\beta > 1$, $\xi > 0$, and $1/2 \leq \eta < 1$ such that for every $z \geq 1$ and every integer $n \geq 1$,

$$\mathbb{P}\big(n\bar{X}_{i,n} - n\mu_i \geq n^{\eta} z\big) \leq \frac{\beta}{z^{\xi}}, \quad \mathbb{P}\big(n\bar{X}_{i,n} - n\mu_i \leq -n^{\eta} z\big) \leq \frac{\beta}{z^{\xi}}. \tag{2.13}$$

### ■ 2.5.1 Algorithm

Consider applying the following variant of Upper Confidence Bound (UCB) algorithm to the above non-stationary MAB. Define upper confidence bound for arm or action $i$ when it is played $s$ times in total of $t \geq s$ time steps as

$$U_{i,s,t} = \bar{X}_{i,s} + B_{t,s}, \tag{2.14}$$

where $B_{t,s}$ is the bonus term. Denote by $I_t$ the arm played at time $t \geq 1$. Then,

$$I_t \in \arg\max_{i\in[K]} \big\{\bar{X}_{i,T_i(t-1)} + B_{t-1,T_i(t-1)}\big\}, \tag{2.15}$$

where $T_i(t) = \sum_{l=1}^{t} \mathbb{I}\{I_l = i\}$ is the number of times arm $i$ has been played, up to (including) time $t$. We shall make specific selection of the bonus term $B_{t,s}$ as

$$B_{t,s} = \frac{\beta^{1/\xi} \cdot t^{\alpha/\xi}}{s^{1-\eta}}. \tag{2.16}$$

A tie is broken arbitrarily when selecting an arm. In the above, $\alpha > 0$ is a tuning parameter that controls the exploration and exploitation trade-off. Let $\mu_* = \max_{i\in[K]} \mu_i$ be the optimal value with respect to the converged expectation, and $i_* \in \arg\max_{i\in[K]} \mu_i$ be the corresponding optimal arm. We assume that the optimal arm is unique. Let $\delta_{i*,n} = \mu_{i_*,n} - \mu_{i_*}$, which measures how fast the mean of the optimal non-stationary arm converges. For simplicity, quantities related to the optimal arm $i_*$ will be simply denoted with subscript $*$, e.g., $\delta_{*,n} = \delta_{i_*,n}$. Finally, denote by $\Delta_{\min} = \min_{i\in[K], i\neq i_*} \Delta_i$ the gap between the optimal arm and the second optimal arm with notation $\Delta_i = \mu_* - \mu_i$.

### ■ 2.5.2 Analysis

Let $\bar{X}_n \triangleq \frac{1}{n}\sum_{i=1}^{K} T_i(n)\bar{X}_{i,T_i(n)}$ denote the empirical average under the UCB algorithm (2.15). Then, $\bar{X}_n$ satisfies the following convergence and concentration properties.

**Theorem 3.** *Consider a non-stationary MAB satisfying (2.12) and (2.13). Suppose that algorithm (2.15) is applied with parameter $\alpha$ such that $\xi\eta(1-\eta) \le \alpha < \xi(1-\eta)$ and $\alpha > 2$. Then, the following holds:*

*A. Convergence:*

$$\left|\mathbb{E}[\bar{X}_n] - \mu_*\right| \le |\delta_{*,n}| + \frac{2R(K-1)\cdot\left(\left(\frac{2}{\Delta_{\min}}\cdot\beta^{1/\xi}\right)^{\frac{1}{1-\eta}}\cdot n^{\frac{\alpha}{\xi(1-\eta)}} + \frac{2}{\alpha-2} + 1\right)}{n}.$$

*B. Concentration: there exist constants, $\beta' > 1$ and $\xi' > 0$ and $1/2 \le \eta' < 1$ such that for every $n \ge 1$ and every $z \ge 1$,*

$$\mathbb{P}\left(n\bar{X}_n - n\mu_* \ge n^{\eta'} z\right) \le \frac{\beta'}{z^{\xi'}}, \quad \mathbb{P}\left(n\bar{X}_n - n\mu_* \le -n^{\eta'} z\right) \le \frac{\beta'}{z^{\xi'}},$$

*where $\eta' = \frac{\alpha}{\xi(1-\eta)}$, $\xi' = \alpha - 1$, $\beta'$ depends on $R, K, \Delta_{\min}, \beta, \xi, \alpha, \eta$.*

## ■ 2.6 Proof of Theorem 3

We establish the convergence and concentration properties of the variant of the Upper Confidence Bound algorithm described in Section 2.5 and specified through (2.14), (2.15) and (2.16).

### ■ 2.6.1 Establishing the Convergence Property

We define a useful notation

$$\Phi(n,\delta) = n^{\eta}\left(\frac{\beta}{\delta}\right)^{1/\xi}. \tag{2.17}$$

We begin with a helpful lemma, which shows that the probability that a non-optimal arm or action has a large upper confidence is polynomially small. Proof is provided in Section 2.6.3.

**Lemma 1.** *Let $i \in [K], i \neq i_*$ be a sub-optimal arm and define*

$$A_i(t) \triangleq \min_{u\in\mathbb{N}}\left\{\frac{\Phi(u,t^{-\alpha})}{u} \leq \frac{\Delta_i}{2}\right\} = \left\lceil\left(\frac{2}{\Delta_i}\cdot\beta^{1/\xi}\cdot t^{\alpha/\xi}\right)^{\frac{1}{1-\eta}}\right\rceil. \tag{2.18}$$

*For each $s$ and $t$ such that, $A_i(t) \leq s \leq t$, we have*

$$\mathbb{P}(U_{i,s,t} > \mu_*) \leq t^{-\alpha}.$$

Lemma 1 implies that as long as each arm is played enough, the sub-optimal ones become less likely to be selected. This allows us to upper bound the expected number of sub-optimal plays as follows.

**Lemma 2.** *Let $i \in [K], i \neq i_*$, then*

$$\mathbb{E}[T_i(n)] \leq \left(\frac{2}{\Delta_i}\cdot\beta^{1/\xi}\right)^{\frac{1}{1-\eta}}\cdot n^{\frac{\alpha}{\xi(1-\eta)}} + \frac{2}{\alpha-2} + 1.$$

Proof of Lemma 2 is deferred to Section 2.6.3.

*Completing Proof of Convergence.* By the triangle inequality,

$$\left|\mu_* - \mathbb{E}[\bar{X}_n]\right| = \left|\mu_* - \mu_{*,n}\right| + \left|\mu_{*,n} - \mathbb{E}[\bar{X}_n]\right| = \left|\delta_{*,n}\right| + \left|\mu_{*,n} - \mathbb{E}[\bar{X}_n]\right|.$$

The second term can be bounded as follows:

$$\begin{aligned}
&n\left|\mu_{*,n} - \mathbb{E}[\bar{X}_n]\right| \\
&= \left|\mathbb{E}\left[\sum_{t=1}^{n} X_{i_*,t}\right] - \mathbb{E}\left[\sum_{i=1}^{K} T_i(n)\bar{X}_{i,T_i(n)}\right]\right| \\
&\leq \left|\mathbb{E}\left[\sum_{t=1}^{n} X_{i_*,t}\right] - \mathbb{E}\left[T_*(n)\bar{X}_{i_*,T_*(n)}\right]\right| + \left|\mathbb{E}\left[\sum_{i=1,i\neq i_*}^{K} T_i(n)\bar{X}_{i,T_i(n)}\right]\right| \\
&= \left|\mathbb{E}\left[\sum_{t=T_*(n)+1}^{n} X_{i_*,t}\right]\right| + \left|\mathbb{E}\left[\sum_{i=1,i\neq i_*}^{K} T_i(n)\bar{X}_{i,T_i(n)}\right]\right|.
\end{aligned} \tag{2.19}$$

Recall that the reward sequences are assumed to be bounded in $[-R, R]$. Therefore, the first term of (2.19) can be bounded as follows:

$$\left|\mathbb{E}\Big[\sum_{t=T_*(n)+1}^{n} X_{i_*,t}\Big]\right| \le \mathbb{E}\Big[\sum_{t=T_*(n)+1}^{n} |X_{i_*,t}|\Big] \le R\cdot\mathbb{E}\Big[\sum_{i=1,i\neq i_*}^{K} T_i(n)\Big].$$

The second term can also be bounded as:

$$\left|\mathbb{E}\Big[\sum_{i=1,i\neq i_*}^{K} T_i(n)\bar{X}_{i,T_i(n)}\Big]\right| \le R\cdot\mathbb{E}\Big[\sum_{i=1,i\neq i_*}^{K} T_i(n)\Big].$$

Hence, we obtain that

$$\big|\mu_* - \mathbb{E}[\bar{X}_n]\big| = |\delta_{*,n}| + \big|\mu_{*,n} - \mathbb{E}[\bar{X}_n]\big| \le |\delta_{*,n}| + \frac{2R\cdot\mathbb{E}\big[\sum_{i=1,i\neq i_*}^{K} T_i(n)\big]}{n}.$$

Combining the above bounds and Lemma 2 yields the desired convergence result in Theorem 3. □

## ■ 2.6.2 Establishing the Concentration Property

Having proved the convergence property, the next step is to show that a similar concentration property (cf. (2.13)) also holds for $\bar{X}_n$. We aim to precisely capture the relationship between the original constants assumed in the assumption and the new constants obtained for $\bar{X}_n$. To begin with, recall the definition of $A_i(t)$ in Lemma 1 and define

$$A(t) = \max_{i\in[K]} A_i(t) = \left\lceil \Big(\frac{2}{\Delta_{\min}}\cdot\beta^{1/\xi}\Big)^{\frac{1}{1-\eta}}\cdot t^{\frac{\alpha}{\xi(1-\eta)}}\right\rceil. \tag{2.20}$$

It can be checked that replacing $\beta$ with any larger number still makes the concentration inequalities (2.13) hold. Without loss of generality, we hence let $\beta$ be large enough so that $\frac{2}{\Delta_{\min}}\cdot\beta^{1/\xi} > 1$. We further denote by $N_p$ the first time such that $t \ge A(t)$, i.e.,

$$N_p = \min\{t\ge 1\ :\ t\ge A(t)\} \ = \ \Theta\Big(\big(\frac{2^\xi\beta}{\Delta_{\min}^\xi}\big)^{\frac{1}{\xi(1-\eta)-\alpha}}\Big). \tag{2.21}$$

We first state the following concentration property, which will be further refined to match the desired form in Theorem 3. We defer the proof to Section 2.6.3.

**Lemma 3.** *For any $n \ge N_p$ and $x \ge 1$, let $r_0 = n^\eta + 2R(K-1)\big(3 + A(n)\big)$. Then,*

$$\mathbb{P}\Big(n\bar{X}_n - n\mu_* \ge r_0 x\Big) \le \frac{\beta}{x^\xi} + \frac{2(K-1)}{(\alpha-1)\big((1+A(n))x\big)^{\alpha-1}},$$

$$\mathbb{P}\Big(n\bar{X}_n - n\mu_* \le -r_0 x\Big) \le \frac{\beta}{x^\xi} + \frac{2(K-1)}{(\alpha-1)\big((1+A(n))x\big)^{\alpha-1}}.$$

Lemma 3 confirms that indeed, as $n$ becomes large, the average $\bar{X}_n$ also satisfies certain concentration inequalities. However, the particular form of concentration in Theorem 3 does not quite match the form of concentration in Theorem 3 which we conclude next.

*Completing Proof of Concentration Property.* Let $N_p'$ be a constant defined as follows:

$$N_p' = \min\{t \ge 1 : t \ge A(t) \text{ and } 2RA(t) \ge t^\eta + 2R(4K-3)\}.$$

Recall the definition of $A(t)$ and that $\alpha \ge \xi\eta(1-\eta)$ and $\alpha < \xi(1-\eta)$. Hence, $N_p'$ is guaranteed to exist. In addition, note that by definition, $N_p' \ge N_p$. For each $n \ge N_p'$,

$$\begin{aligned}
2RK\Big(\frac{2}{\Delta_{\min}}\cdot\beta^{1/\xi}\Big)^{\frac{1}{1-\eta}}\cdot n^{\frac{\alpha}{\xi(1-\eta)}} &= 2RK\Big[\Big(\frac{2}{\Delta_{\min}}\cdot\beta^{1/\xi}\Big)^{\frac{1}{1-\eta}}\cdot n^{\frac{\alpha}{\xi(1-\eta)}} + 1 - 1\Big] \\
&\ge 2RKA(n) - 2RK \\
&= 2R(K-1)A(n) + 2RA(n) - 2RK \\
&\ge 2R(K-1)A(n) + n^\eta + 2R(4K-3) - 2RK \\
&= 2R(K-1)(A(n)+3) + n^\eta = r_0
\end{aligned}$$

Now, let us apply Lemma 3: for every $n \ge N_p'$ and $x \ge 1$, we have

$$\begin{aligned}
\mathbb{P}\Big(n\bar{X}_n - n\mu_* \ge n^{\frac{\alpha}{\xi(1-\eta)}}\Big[2RK\Big(\frac{2}{\Delta_{\min}}\cdot\beta^{1/\xi}\Big)^{\frac{1}{1-\eta}}\Big]x\Big) &\le \mathbb{P}\Big(n\bar{X}_n - n\mu_* \ge r_0 x\Big) \\
&\le \frac{\beta}{x^\xi} + \frac{2(K-1)}{(\alpha-1)\big((1+A(n))x\big)^{\alpha-1}} \\
&\le \frac{2\max(\beta, \frac{2(K-1)}{(\alpha-1)(1+A(N_p'))^{\alpha-1}})}{x^{\alpha-1}}, \quad (2.22)
\end{aligned}$$

where the last inequality follows because $n \ge N_p'$ and $A(n)$ is a non-decreasing function. In addition, since $\alpha < \xi(1-\eta) < \xi$, we have $\alpha - 1 < \xi$. For convenience, we define a constant

$$c_1 \triangleq 2RK\Big(\frac{2}{\Delta_{\min}}\cdot\beta^{1/\xi}\Big)^{\frac{1}{1-\eta}}. \quad (2.23)$$

Equivalently, by a change of variable, i.e., letting $z = c_1 x$, then for every $n \ge N_p'$ and $z \ge 1$, we obtain that

$$\mathbb{P}\Big(n\bar{X}_n - n\mu_* \ge n^{\frac{\alpha}{\xi(1-\eta)}} z\Big) \le \frac{2c_1^{\alpha-1}\cdot\max\big(\beta, \frac{2(K-1)}{(\alpha-1)(1+A(N_p'))^{\alpha-1}}\big)}{z^{\alpha-1}}. \quad (2.24)$$

The above inequality holds because: (1) if $z \geq c_1$, then (2.24) directly follows from (2.22); (2) if $1 \leq z \leq c_1$, then the R.H.S. of (2.24) is at least 1 (by assumption, $\beta > 1$) and the inequality trivially holds. The concentration inequality, i.e., (2.24), is now almost the same as the desired form in Theorem 3. The only difference is that it only holds for $n \geq N_p'$. This is not hard to resolve. The easiest approach, which we show in the following, is to refine the constants to ensure that when $1 \leq n < N_p'$, (2.24) is trivially true. To this end, we note that $|n\bar{X}_n - n\mu_*| \leq 2Rn$. For each $1 \leq n < N_p'$, there is a corresponding $\bar{z}(n)$ such that $n^{\frac{\alpha}{\xi(1-\eta)}}\bar{z}(n) = 2Rn$. That is,

$$\bar{z}(n) \triangleq 2Rn^{1-\frac{\alpha}{\xi(1-\eta)}}, \quad 1 \leq n < N_p'.$$

This then implies that for each $1 \leq n < N_p'$, the following inequality trivially holds:

$$\mathbb{P}\Big(n\bar{X}_n - n\mu_* \geq n^{\frac{\alpha}{\xi(1-\eta)}} z\Big) \leq \frac{\bar{z}(n)^{\alpha-1}}{z^{\alpha-1}}, \quad \forall\, z \geq 1.$$

To see why, note that for each $1 \leq n < N_p'$: (1) if $z \geq \bar{z}(n)$, then $n^{\frac{\alpha}{\xi(1-\eta)}} z \geq 2Rn$ and the above probability should be 0. Hence, any positive number on the R.H.S. makes the inequality trivially true; (2) if $1 \leq z < \bar{z}(n)$, the R.H.S. is at least 1, which again makes the inequality hold. For convenience, define

$$c_2 \triangleq \max_{1 \leq n < N_p'} \bar{z}(n) = 2R(N_p' - 1)^{1-\frac{\alpha}{\xi(1-\eta)}}. \tag{2.25}$$

Then, it is easy to see that for every $n \geq 1$ and every $z \geq 1$, we have

$$\mathbb{P}\big(n\bar{X}_n - n\mu_* \geq n^{\eta'} z\big) \leq \frac{\beta'}{z^{\xi'}},$$

where the constants are given by

$$\eta' = \frac{\alpha}{\xi(1-\eta)}, \tag{2.26}$$

$$\xi' = \alpha - 1, \tag{2.27}$$

$$\beta' = \max\Big\{c_2,\ 2c_1^{\alpha-1} \cdot \max\big(\beta, \frac{2(K-1)}{(\alpha-1)(1+A(N_p'))^{\alpha-1}}\big)\Big\}. \tag{2.28}$$

Finally, notice that since $\alpha \geq \xi\eta(1-\eta)$ and $\alpha < \xi(1-\eta)$, we have $1/2 \leq \eta \leq \eta' < 1$. Note that per (2.23), $c_1$ depends on $R, K, \Delta_{\min}, \beta, \xi$ and $\eta$. In addition, $c_2$ depends on $R, K, \Delta_{\min}, \beta, \xi, \alpha, \eta$ and $N_p'$ depends on $R, K, \Delta_{\min}, \beta, \xi, \alpha, \eta$. Therefore, $\beta'$ depends on $R, K, \Delta_{\min}, \beta,\ \xi, \alpha, \eta$. The other direction follows exactly the same reasoning, and this completes the proof of Theorem 3. □

### ■ 2.6.3 Proofs of Lemmas 1, 2 & 3

**Proof of Lemma 1.** By the choice of $A_i(t)$, $s$ and $t$, we have $B_{t,s} = \frac{\Phi(s,t^{-\alpha})}{s} \le \frac{\Phi(A_i(t),t^{-\alpha})}{A_i(t)} \le \frac{\Delta_i}{2}$. Therefore,

$$\begin{aligned}
\mathbb{P}(U_{i,s,t} > \mu_*) &= \mathbb{P}(\bar{X}_{i,s} + B_{t,s} > \mu_*) \\
&= \mathbb{P}\Big(\bar{X}_{i,s} - \mu_i > \Delta_i - B_{t,s}\Big) \\
&\le \mathbb{P}\Big(\bar{X}_{i,s} - \mu_i > B_{t,s}\Big) && \Delta_i \ge 2B_{t,s} \\
&\le t^{-\alpha}. && \text{by concentration (2.13).}
\end{aligned}$$

□

**Proof of Lemma 2.** If a sub-optimal arm $i$ is chosen at time $t+1$, i.e., $I_{t+1} = i$, then at least one of the following two equations must be true: with notation $T_*(\cdot) = T_{i_*}(\cdot)$,

$$U_{i_*,T_*(t),t} \le \mu_* \,, \tag{2.29}$$

$$U_{i,T_i(t),t} > \mu_* \,. \tag{2.30}$$

Indeed, if both inequalities are false, we have $U_{i_*,T_*(t),t} > \mu_* \ge U_{i,T_i(t),t}$, which is a contradiction to $I_{t+1} = i$. We now use this fact to prove Lemma 2.

**Case 1:** $n > A_i(n)$. Note that such $n$ exists because $A_i(n)$ grows with a polynomial order $O\big(n^{\frac{\alpha}{\xi(1-\eta)}}\big)$ and $\alpha < \xi(1-\eta)$, i.e., $A_i(n) = o(n)$. Then,

$$\begin{aligned}
T_i(n) &= \sum_{t=0}^{n-1} \mathbb{I}\{I_{t+1} = i\} \overset{(a)}{=} 1 + \sum_{t=K}^{n-1} \mathbb{I}\{I_{t+1} = i\} \\
&= 1 + \sum_{t=K}^{n-1} \big(\mathbb{I}\{I_{t+1} = i, T_i(t) < A_i(n)\} + \mathbb{I}\{I_{t+1} = i, T_i(t) \ge A_i(n)\}\big) \\
&\le A_i(n) + \sum_{t=K}^{n-1} \mathbb{I}\{I_{t+1} = i, T_i(t) \ge A_i(n)\},
\end{aligned}$$

where equality (a) follows from the fact that $B_{t,s} = \infty$ if $s = 0$.

To analyze the above summation, we note that from (2.29) and (2.30),

$$\begin{aligned}
&\mathbb{I}\{I_{t+1} = i, T_i(t) \ge A_i(n)\} \\
&\le \mathbb{I}\{U_{i_*,T_*(t),t} \le \mu_* \text{ or } U_{i,T_i(t),t} > \mu_*, T_i(t) \ge A_i(n)\} \\
&\le \mathbb{I}\{U_{i,T_i(t),t} > \mu_*, T_i(t) \ge A_i(n)\} + \mathbb{I}\{U_{i_*,T_*(t),t} \le \mu_*, T_i(t) \ge A_i(n)\}
\end{aligned}$$

$$\leq \mathbb{I}\{U_{i,T_i(t),t} > \mu_*, T_i(t) \geq A_i(n)\} + \mathbb{I}\{U_{i_*,T_*(t),t} \leq \mu_*\}$$
$$= \mathbb{I}\{\exists\, s : A_i(n) \leq s \leq t, \text{ s.t. } U_{i,s,t} > \mu_*\} + \mathbb{I}\{\exists\, s_* : 1 \leq s_* \leq t, \text{ s.t. } U_{i_*,s_*,t} \leq \mu_*\}.$$

To summarize, we have proved that

$$\begin{aligned}
&\mathbb{E}[T_i(n)] \\
&\leq A_i(n) + \sum_{t=A_i(n)}^{n-1} \mathbb{P}\Big((2.29) \text{ or } (2.30) \text{ is true, and } T_i(t) \geq A_i(n)\Big) \\
&\leq A_i(n) + \sum_{t=A_i(n)}^{n-1} \Big[\mathbb{P}\big(\underbrace{\exists\, s : A_i(n) \leq s \leq t, \text{ s.t. } U_{i,s,t} > \mu_*}_{E_1}\big) + \mathbb{P}\big(\underbrace{\exists\, s_* : 1 \leq s_* \leq t, \text{ s.t. } U_{i_*,s_*,t} \leq \mu_*}_{E_2}\big)\Big].
\end{aligned} \tag{2.31}$$

To complete the proof of Lemma 2, it suffices to bound the probabilities of the two events $E_1$ and $E_2$. To this end, we use a union bound:

$$\mathbb{P}(E_1) \leq \sum_{s=A_i(n)}^{t} \mathbb{P}\big(U_{i,s,t} > \mu_*\big) \overset{(a)}{\leq} \sum_{s=A_i(n)}^{t} t^{-\alpha} \leq t \cdot t^{-\alpha} = t^{1-\alpha},$$

where step $(a)$ follows from $A_i(n) \geq A_i(t)$ and Lemma 1. We bound $\mathbb{P}(E_2)$ in a similar way:

$$\mathbb{P}(E_2) \leq \sum_{s_*=1}^{t} \mathbb{P}(U_{i_*,s_*,t} \leq \mu_*) = \sum_{s_*=1}^{t} \mathbb{P}\left(\bar{X}_{i_*,s_*} + B_{t,s_*} \leq \mu_*\right) \overset{(a)}{\leq} \sum_{s_*=1}^{t} t^{-\alpha} \leq t^{1-\alpha},$$

where step $(a)$ follows from concentration (cf. (2.13)). By substituting the bounds of $\mathbb{P}(E_1)$ and $\mathbb{P}(E_2)$ into (2.31), we have:

$$\begin{aligned}
\mathbb{E}[T_i(n)] &\leq A_i(n) + \sum_{t=A_i(n)}^{n-1} 2t^{1-\alpha} \\
&\leq A_i(n) + \int_{A_i(n)-1}^{\infty} 2t^{1-\alpha} dt && \alpha > 2 \\
&= A_i(n) + \frac{2\big(A_i(n)-1\big)^{2-\alpha}}{\alpha - 2} \\
&\leq A_i(n) + \frac{2}{\alpha - 2} \\
&\leq \left(\frac{2}{\Delta_i} \cdot \beta^{1/\xi}\right)^{\frac{1}{1-\eta}} \cdot n^{\frac{\alpha}{\xi(1-\eta)}} + \frac{2}{\alpha - 2} + 1.
\end{aligned}$$

**Case 2:** $n \leq A_i(n)$. Note that if $n$ is such that $n \leq A_i(n)$, then the above bound trivially

holds because $T_i(n) \le n \le A_i(n)$. This completes the proof of Lemma 2. □

**Proof of Lemma 3.** We first prove one direction, namely, $\mathbb{P}(n\mu_* - n\bar{X}_n \ge r_0 x)$. The other direction follows the similar steps, and we will comment on that at the end of this proof. The general idea underlying the proof is to rewrite the quantity $n\mu_* - n\bar{X}_n$ as sums of terms that can be bounded using previous lemmas or assumptions. To begin with, note that

$$\begin{aligned} n\mu_* - n\bar{X}_n &= n\mu_* - \sum_{i=1}^{K} T_i(n)\bar{X}_{i,T_i(n)} \\ &= n\mu_* - \sum_{t=1}^{T_*(n)} X_{i_*,t} - \sum_{i\neq i_*} T_i(n)\bar{X}_{i,T_i(n)} \\ &= n\mu_* - \sum_{t=1}^{n} X_{i_*,t} + \sum_{t=T_*(n)+1}^{n} X_{i_*,t} - \sum_{i\neq i_*}\sum_{t=1}^{T_i(n)} X_{i,t} \\ &\le n\mu_* - \sum_{t=1}^{n} X_{i_*,t} + 2R\sum_{i\neq i_*} T_i(n), \end{aligned}$$

because $X_{i,t} \in [-R, R]$ for all $i, t$. Therefore, we have

$$\begin{aligned} \mathbb{P}\big(n\mu_* - n\bar{X}_n \ge r_0 x\big) &\le \mathbb{P}\big(n\mu_* - \sum_{t=1}^{n} X_{i_*,t} + 2R\sum_{i\neq i_*} T_i(n) \ge r_0 x\big) \\ &\le \mathbb{P}\big(n\mu_* - \sum_{t=1}^{n} X_{i_*,t} \ge n^{\eta} x\big) + \sum_{i\neq i_*} \mathbb{P}\big(T_i(n) \ge (3 + A(n))x\big), \end{aligned} \tag{2.32}$$

where the last inequality follows from the union bound.

To prove the theorem, we now bound the two terms in (2.32). By our concentration assumption, we can upper bound the first term as follows:

$$\mathbb{P}\big(n\mu_* - \sum_{t=1}^{n} X_{i_*,t} \ge n^{\eta} x\big) \le \frac{\beta}{x^{\xi}}. \tag{2.33}$$

Next, we bound each term in the summation of (2.32). Fix $n$ and a sub-optimal arm $i$. Let $u$ be an integer satisfying $u \ge A(n)$. For any $\tau \in \mathbb{R}$, consider the following two events:

$$\begin{aligned} E_1 &= \big\{\text{For each integer } t \in [u, n],\ \text{we have } U_{i,u,t} \le \tau\big\}, \\ E_2 &= \big\{\text{For each integer } s \in [1, n-u],\ \text{we have } U_{i_*,s,u+s} > \tau\big\}. \end{aligned}$$

As a first step, we want to show that

$$E_1 \cap E_2 \Rightarrow T_i(n) \leq u. \tag{2.34}$$

To this end, let us condition on both events $E_1$ and $E_2$. Recall that $B_{t,s}$ is non-decreasing with respect to $t$. Then, for each $s$ such that $1 \leq s \leq n-u$, and each $t$ such that $u+s \leq t \leq n$, it holds that

$$U_{i_*,s,t} = \bar{X}_{i_*,s} + B_{t,s} \geq \bar{X}_{i_*,s} + B_{u+s,s} = U_{i_*,s,u+s} > \tau \geq U_{i,u,t}.$$

This implies that $T_i(n) \leq u$. To see why, suppose that $T_i(n) > u$ and denote by $t'$ the first time that arm $i$ has been played $u$ times, i.e., $t' = \min\{t : t \leq n, T_i(t) = u\}$. Note that by definition $t' \geq u + T_*(t')$. Hence, for any time t such that $t' < t \leq n$, the above inequality implies that $U_{i_*,T_*(t),t} > U_{i,u,t}$. That is, $i^*$ always has a higher upper confidence bound than $i$, and arm $i$ will not be selected, i.e., arm $i$ will not be played the $(u+1)$-th time. This contradicts our assumption that $T_i(n) > u$, and hence we have the inequality $T_i(n) \leq u$.

To summarize, we have established the fact that $E_1 \cap E_2 \Rightarrow T_i(n) \leq u$. As a result, we have:

$$\begin{aligned}\{T_i(n) > u\} &\subset \big(E_1^c \cup E_2^c\big) \\ &= \Big(\big\{\exists\, t :\ u \leq t \leq n \text{ s.t. } U_{i,u,t} > \tau\big\} \cup \big\{\exists\, s : 1 \leq s \leq n-u, \text{ s.t. } U_{i_*,s,u+s} \leq \tau\big\}\Big).\end{aligned}$$

Using union bound, we obtain that

$$\mathbb{P}\big(T_i(n) > u\big) \leq \sum_{t=u}^{n} \mathbb{P}(U_{i,u,t} > \tau) + \sum_{s=1}^{n-u} \mathbb{P}\big(U_{i_*,s,u+s} \leq \tau\big). \tag{2.35}$$

Note that for the above bound, we are free to choose $u$ and $\tau$ as long as $u \geq A(n)$. To connect with our goal (cf. (2.32)), in the following, we set $u = \lfloor (1+A(n))x \rfloor + 1$ (recall that $x \geq 1$) and $\tau = \mu_*$ to bound $\mathbb{P}(T_i(n) > u)$. Since $u \geq A(n) \geq A_i(n)$, by Lemma 1, we have

$$\begin{aligned}\sum_{t=u}^{n} \mathbb{P}(U_{i,u,t} > \mu_*) \leq \sum_{t=u}^{n} t^{-\alpha} \leq \int_{u-1}^{\infty} t^{-\alpha} dt &= \frac{(u-1)^{1-\alpha}}{\alpha - 1} \\ &= \frac{(\lfloor (1+A(n))x \rfloor)^{1-\alpha}}{\alpha - 1} \leq \frac{\Big((1+A(n))x\Big)^{1-\alpha}}{\alpha - 1}.\end{aligned}$$

As for the second summation in the R.H.S. of (2.35), we have that

$$\sum_{s=1}^{n-u} \mathbb{P}\big(U_{i_*,s,u+s} \leq \tau\big) = \sum_{s=1}^{n-u} \mathbb{P}(U_{i_*,s,u+s} \leq \mu_*)$$

$$
\begin{aligned}
&= \sum_{s=1}^{n-u} \mathbb{P}\Big( \bar{X}_{i_*,s} + B_{u+s,s} \le \mu_* \Big) \\
&\le \sum_{s=1}^{n-u} (s+u)^{-\alpha} = \sum_{t=1+u}^{n} t^{-\alpha} \\
&\le \int_{u-1}^{\infty} t^{-\alpha} dt = \frac{(u-1)^{1-\alpha}}{\alpha-1} \le \frac{\Big((1+A(n))x\Big)^{1-\alpha}}{\alpha-1},
\end{aligned}
$$

where the first inequality follows from the concentration property, cf. (2.13). Combining the above inequalities and note that $(3+A(n))x > \lfloor (1+A(n))x \rfloor + 1$:

$$
\mathbb{P}\big(T_i(n) \ge (3+A(n))x\big) \le \mathbb{P}\big(T_i(n) > u\big) \le \frac{2\Big(\big(1+A(n)\big)x\Big)^{1-\alpha}}{\alpha-1}. \tag{2.36}
$$

Substituting (2.33) and (2.36) into (2.32), we obtain

$$
\mathbb{P}\big(n\mu_* - n\bar{X}_n \ge r_0 x\big) \le \frac{\beta}{x^{\xi}} + \sum_{i \ne i_*} \frac{2\Big(\big(1+A(n)\big)x\Big)^{1-\alpha}}{\alpha-1},
$$

which is the desired inequality in Lemma 3.

To complete the proof, we need to consider the other direction, i.e., $\mathbb{P}(n\bar{X}_n - n\mu_* \ge r_0 x)$. The proof is almost identical. Note that

$$
\begin{aligned}
n\bar{X}_n - n\mu_* &= \sum_{i=1}^{K} T_i(n) \bar{X}_{i,T_i(n)} - n\mu_* \\
&= \sum_{t=1}^{n} X_{i_*,t} - n\mu_* - \sum_{t=T_*(n)+1}^{n} X_{i_*,t} + \sum_{i \ne i_*} \sum_{t=1}^{T_i(n)} X_{i,t} \\
&\le \sum_{t=1}^{n} X_{i_*,t} - n\mu_* + 2R \sum_{i \ne i_*} T_i(n),
\end{aligned}
$$

because $X_{i,t} \in [-R, R]$ for all $i, t$. Therefore,

$$
\begin{aligned}
\mathbb{P}\big(n\bar{X}_n - n\mu_* \ge r_0 x\big) &\le \mathbb{P}\Big( \sum_{t=1}^{n} X_{i_*,t} - n\mu_* + 2R \sum_{i \ne i_*} T_i(n) \ge r_0 x \Big) \\
&\le \mathbb{P}\Big( \sum_{t=1}^{n} X_{i_*,t} - n\mu_* \ge n^{\eta} x \Big) + \sum_{i \ne i_*} \mathbb{P}\big(T_i(n) \ge (3+A_i(n))x\big).
\end{aligned}
$$

The desired inequality then follows exactly from the same reasoning as before. □

## ■ 2.7 Analysis of MCTS and Proof of Theorem 1

With the previous results for the non-stationary MAB, in this section, we give a complete analysis for the fixed-depth Monte Carlo Tree Search (MCTS) algorithm illustrated in Algorithm 1 and prove Theorem 1. In effect, as discussed in Section 2.3, one can view a depth-$H$ MCTS as a simulated version of $H$ steps value function iteration. Given the current value function proxy $\hat{V}$, let $V^{(H)}(\cdot)$ be the value function estimation after $H$ steps of value function iteration starting with the proxy $\hat{V}$. Then, we prove the result in two parts. First, we argue that due to the MCTS sampling process, the mean of the empirical estimation of value function at the query node $s$, or the root node of MCTS tree, is within $O(n^{\eta-1})$ of $V^{(H)}(s)$ after $n$ simulations, with the given proxy $\hat{V}$ being the input to the MCTS algorithm. Second, we argue that $V^{(H)}(s)$ is within $\gamma^H\|\hat{V}-V^*\|_\infty \le \gamma^H\varepsilon_0$ of the optimal value function. Putting this together leads to Theorem 1.

We start by a preliminary probabilistic lemma in Section 2.7.1 that will be useful throughout. Sections 2.7.2 and 2.7.3 argue the first part of the proof as explained above. Section 2.7.4 provides proof of the second part. And Section 2.7.5 concludes the proof of Theorem 1.

### ■ 2.7.1 Preliminary

We state the following probabilistic lemma that is useful throughout. Proof can be found in Section 2.7.6.

**Lemma 4.** *Consider real-valued random variables $X_i, Y_i$ for $i \ge 1$ such that $X$s are independent and identically distributed taking values in $[-B, B]$ for some $B > 0$, $X$s are independent of $Y$s, and $Y$s satisfy*

*A. Convergence: for $n \ge 1$, with notation $\bar{Y}_n = \frac{1}{n}\big(\sum_{i=1}^n Y_i\big)$,*

$$\lim_{n\to\infty} \mathbb{E}[\bar{Y}_n] = \mu_Y.$$

*B. Concentration: there exist constants, $\beta > 1$, $\xi > 0$, $1/2 \le \eta < 1$ such that for $n \ge 1$ and $z \ge 1$,*

$$\mathbb{P}\big(n\bar{Y}_n - n\mu_Y \ge n^\eta z\big) \le \frac{\beta}{z^\xi}, \quad \mathbb{P}\big(n\bar{Y}_n - n\mu_Y \le -n^\eta z\big) \le \frac{\beta}{z^\xi}.$$

*Let $Z_i = X_i + \rho Y_i$ for some $\rho > 0$. Then, $Z$s satisfy*

*A. Convergence: for $n \ge 1$, with notation $\bar{Z}_n = \frac{1}{n}\big(\sum_{i=1}^n Z_i\big)$, and $\mu_X = \mathbb{E}[X_1]$,*

$$\lim_{n\to\infty} \mathbb{E}[\bar{Z}_n] = \mu_X + \rho\mu_Y.$$

*B. Concentration: there exist constant $\beta' > 1$ depending upon $\rho, \xi, \beta$ and $B$, such that for $n \geq 1$ and $z \geq 1$,*

$$\mathbb{P}\big(n\bar{Z}_n - n(\mu_X + \rho\mu_Y) \geq n^\eta z\big) \leq \frac{\beta'}{z^\xi}, \quad \mathbb{P}\big(n\bar{Z}_n - n(\mu_X + \rho\mu_Y) \leq -n^\eta z\big) \leq \frac{\beta'}{z^\xi}.$$

## ■ 2.7.2 Analyzing Leaf Level $H$

The goal is to understand the empirical reward observed at the query node for MCTS or the root node of the MCTS tree. In particular, we argue that the mean of the empirical reward at the root node of the MCTS tree is within $O(n^{\eta-1})$ of the mean reward obtained at it assuming access to infinitely many samples. We start by analyzing the reward collected at the nodes that are at leaf level $H$ and level $H-1$.

The nodes at leaf level, i.e., level $H$, are children of nodes at level $H-1$ in the MCTS tree. Let there be $n_{H-1}$ nodes at level $H-1$ corresponding to states $s_{1,H-1}, \ldots, s_{n_{H-1},H-1} \in \mathcal{S}$. Consider node $i \in [n_{H-1}]$ at level $H-1$, corresponding to state $s_{i,H-1}$. As part of the algorithm, whenever this node is visited, one of the $K$ feasible actions is taken. When an action $a \in [K]$ is taken, the node $s'_H = s_{i,H-1} \circ a$, at the leaf level $H$ is reached. This results in reward at node $s_{i,H-1}$ (at level $H-1$) being equal to $\mathcal{R}(s_{i,H-1}, a) + \gamma\tilde{v}^{(H)}(s'_H)$. Here, for each $s \in \mathcal{S}$ and $a \in [K]$, the reward $\mathcal{R}(s,a)$ is an independent, bounded random variable taking value in $[-R_{\max}, R_{\max}]$ with distribution dependent on $s, a$; $\tilde{v}^{(H)}(\cdot)$ is the input of value function proxy to the MCTS algorithm denoted as $\hat{V}(\cdot)$, and $\gamma \in [0,1)$ is the discounting factor. Recall that $\varepsilon_0 = \|\hat{V} - V^*\|_\infty$ and $\|V^*\|_\infty \leq V_{\max}$. Therefore, $\|\tilde{v}^{(H)}\|_\infty = \|\hat{V}\|_\infty \leq V_{\max} + \varepsilon_0$, and the reward collected at node $s_{i,H-1}$ by following any action is bounded, in absolute value, by $\tilde{R}^{(H-1)}_{\max} = R_{\max} + \gamma(V_{\max} + \varepsilon_0)$.

As part of the MCTS algorithm as described in (2.4), when node $s_{i,H-1}$ is visited for the $t+1$ time with $t \geq 0$, the action taken is

$$\arg\max_{a \in \mathcal{A}} \left\{ \frac{1}{u_a} \sum_{j=1}^{u_a} \Big( r(s_{i,H-1}, a)(j) + \gamma\tilde{v}^{(H)}(s_{i,H-1} \circ a)(j) \Big) + \frac{\big(\beta^{(H)}\big)^{1/\xi^{(H)}} \cdot \big(t\big)^{\alpha^{(H)}/\xi^{(H)}}}{\big(u_a\big)^{1-\eta^{(H)}}} \right\},$$

where $u_a \leq t$ is the number of times action $a$ has been chosen thus far at state $s_{i,H-1}$ in the $t$ visits so far, $r(s_{i,H-1}, a)(j)$ is the $j$th sample of random variable per distribution $\mathcal{R}(s_{i,H-1}, a)$, and $\tilde{v}^{(H)}(s_{i,H-1} \circ a)(j)$ is the reward evaluated at leaf node $s_{i,H-1} \circ a$ for the $j$th time. Note that for all $j$, the reward evaluated at leaf node $s_{i,H-1} \circ a$ is the same and equals to $\tilde{v}^{(H)}(\cdot)$, the input value function proxy for the algorithm. When $u_a = 0$, we use notation $\infty$ to represent quantity inside the arg max. The net discounted reward collected by node $s_{i,H-1}$ during its total of $t \geq 1$ visits is simply the sum of rewards obtained by selecting the actions per the policy – which includes the reward associated with taking an

action and the evaluation of $\tilde{v}^{(H)}(\cdot)$ for appropriate leaf node, discounted by $\gamma$. In effect, at each node $s_{i,H-1}$, we are using the UCB policy described in Section 2.5 with parameters $\alpha^{(H)}, \beta^{(H)}, \xi^{(H)}, \eta^{(H)}$ with $K$ possible actions, where the rewards collected by playing any of these $K$ actions each time is simply the summation of bounded independent and identical (for a given action) random variable and a deterministic evaluation. By applying Lemma 4, where $X$s correspond to independent rewards, $\rho = \gamma$, and $Y$s correspond to deterministic evaluations of $\tilde{v}^{(H)}(\cdot)$, we obtain that for given $\xi^{(H)} > 0$ and $\eta^{(H)} \in [\frac{1}{2}, 1)$, there exists $\beta^{(H)}$ such that the collected rewards at $s_{i,H-1}$ (i.e., sum of i.i.d. reward and deterministic evaluations) satisfy the convergence property (cf. (2.12)) and concentration property (cf. (2.13)) stated in Section 2.5. Therefore, by an application of Theorem 3, we conclude Lemma 5 stated below. We define some notations first:

$$\begin{aligned}
\mu_a^{(H-1)}(s_{i,H-1}) &= \mathbb{E}[\mathcal{R}(s_{i,H-1}, a)] + \gamma \tilde{v}^{(H)}(s_{i,H-1} \circ a), \\
\mu_*^{(H-1)}(s_{i,H-1}) &= \max_{a \in [K]} \mu_a^{(H-1)}(s_{i,H-1}) \\
a_*^{(H-1)}(s_{i,H-1}) &\in \arg\max_{a \in [K]} \mu_a^{(H-1)}(s_{i,H-1}) \\
\Delta_{\min}^{(H-1)}(s_{i,H-1}) &= \mu_*^{(H-1)}(s_{i,H-1}) - \max_{a \neq a_*^{(H-1)}(s_{i,H-1})} \mu_a^{(H-1)}(s_{i,H-1}).
\end{aligned} \tag{2.37}$$

We shall assume that the maximizer in the set $\arg\max_{a \in [K]} \mu_a^{(H-1)}(s_{i,H-1})$ is unique, i.e. $\Delta_{\min}^{(H-1)}(s_{i,H-1}) > 0$. And further note that all rewards belong to $[-\tilde{R}_{\max}^{(H-1)}, \tilde{R}_{\max}^{(H-1)}]$.

**Lemma 5.** *Consider a node corresponding to state $s_{i,H-1}$ at level $H-1$ within the MCTS for $i \in [n_{H-1}]$. Let $\tilde{v}^{(H-1)}(s_{i,H-1})_n$ be the total discounted reward collected at $s_{i,H-1}$ during $n \geq 1$ visits of it, to one of its $K$ leaf nodes under the UCB policy. Then, for the choice of appropriately large $\beta^{(H)} > 0$, for a given $\xi^{(H)} > 0$, $\eta^{(H)} \in [\frac{1}{2}, 1)$ and $\alpha^{(H)} > 2$, we have*

*A. Convergence:*

$$\begin{aligned}
&\left| \mathbb{E}\Big[ \frac{1}{n} \tilde{v}^{(H-1)}(s_{i,H-1})_n \Big] - \mu_*^{(H-1)}(s_{i,H-1}) \right| \\
&\leq \frac{2\tilde{R}_{\max}^{(H-1)}(K-1) \cdot \Big( \big( \frac{2(\beta^{(H)})^{\frac{1}{\xi^{(H)}}}}{\Delta_{\min}^{(H-1)}(s_{i,H-1})} \big)^{\frac{1}{1-\eta^{(H)}}} \cdot n^{\frac{\alpha^{(H)}}{\xi^{(H)}(1-\eta^{(H)})}} + \frac{2}{\alpha^{(H)}-2} + 1 \Big)}{n}.
\end{aligned}$$

*B. Concentration: there exist constants, $\beta' > 1$ and $\xi' > 0$ and $1/2 \leq \eta' < 1$ such that for every $n \geq 1$ and every $z \geq 1$,*

$$\mathbb{P}\big( \tilde{v}^{(H-1)}(s_{i,H-1})_n - n\mu_*^{(H-1)}(s_{i,H-1}) \geq n^{\eta'} z \big) \leq \frac{\beta'}{z^{\xi'}},$$

$$\mathbb{P}\Big(\tilde{v}^{(H-1)}(s_{i,H-1})_n - n\mu_*^{(H-1)}(s_{i,H-1}) \leq -n^{\eta'} z\Big) \leq \frac{\beta'}{z^{\xi'}},$$

*where* $\eta' = \frac{\alpha^{(H)}}{\xi^{(H)}(1-\eta^{(H)})}$, $\xi' = \alpha^{(H)} - 1$, *and* $\beta'$ *is a large enough constant that is function of parameters* $\alpha^{(H)}, \beta^{(H)}, \xi^{(H)}, \eta^{(H)}, \tilde{R}_{\max}^{(H-1)}, K, \Delta_{\min}^{(H-1)}(s_{i,H-1})$.

Let $\Delta_{\min}^{(H-1)} = \min_{i\in[n_{H-1}]} \Delta_{\min}^{(H-1)}(s_{i,H-1})$. Then, the rate of convergence for each node $s_{i,H-1}$, $i \in [n_{H-1}]$ can be uniformly simplified as

$$\begin{aligned}
\delta_n^{(H-1)} &= \frac{2\tilde{R}_{\max}^{(H-1)}(K-1)\cdot\Big(\big(\frac{2(\beta^{(H)})^{\frac{1}{\xi^{(H)}}}}{\Delta_{\min}^{(H-1)}}\big)^{\frac{1}{1-\eta^{(H)}}}\cdot n^{\frac{\alpha^{(H)}}{\xi^{(H)}(1-\eta^{(H)})}} + \frac{2}{\alpha^{(H)}-2} + 1\Big)}{n} \\
&= \Theta\Big(n^{\frac{\alpha^{(H)}}{\xi^{(H)}(1-\eta^{(H)})}-1}\Big) \\
&\overset{(a)}{=} O\big(n^{\eta-1}\big),
\end{aligned}$$

where (a) holds since $\alpha^{(H)} = \xi^{(H)}(1-\eta^{(H)})\eta^{(H)}$, $\eta^{(H)} = \eta$. It is worth remarking that $\mu_*^{(H-1)}(s_{i,H-1})$, as defined in (2.37), is precisely the value function estimation for $s_{i,H-1}$ at the end of one step of value iteration starting with $\hat{V}$.

### ■ 2.7.3 Recursion: Going From Level $h$ to $h-1$

Lemma 5 suggests that the necessary assumption of Theorem 3, i.e. (2.12) and (2.13), is satisfied by $\tilde{v}_n^{(H-1)}$ for each node or state at level $H-1$, with $\alpha^{(H-1)}, \xi^{(H-1)}, \eta^{(H-1)}$ as defined per relationship (2.5) - (2.7) and with appropriately defined large enough constant $\beta^{(H-1)}$. We shall argue that result similar to Lemma 5, but for node at level $H-2$, continues to hold with parameters $\alpha^{(H-2)}, \xi^{(H-2)}, \eta^{(H-2)}$ as defined per relationship (2.5) - (2.7) and with appropriately defined large enough constant $\beta^{(H-2)}$. And similar argument will continue to apply going from level $h$ to $h-1$ for all $h \leq H-1$. That is, we shall assume that the necessary assumption of Theorem 3, i.e. (2.12) and (2.13), holds for $\tilde{v}^{(h)}(\cdot)$, for all nodes at level $h$ with $\alpha^{(h)}, \xi^{(h)}, \eta^{(h)}$ as defined per relationship (2.5) - (2.7) and with appropriately defined large enough constant $\beta^{(h)}$, and then argue that such holds for nodes at level $h-1$ as well. This will, using mathematical induction, allow us to prove the results for all $h \geq 1$.

To that end, consider any node at level $h-1$. Let there be $n_{h-1}$ nodes at level $h-1$ corresponding to states $s_{1,h-1}, \ldots, s_{n_{h-1},h-1} \in \mathcal{S}$. Consider a node corresponding to state $s_{i,h-1}$ at level $h-1$ within the MCTS for $i \in [n_{h-1}]$. As part of the algorithm, whenever this node is visited, one of the $K$ feasible action is taken. When an action $a \in [K]$ is taken, the node $s'_h = s_{i,h-1} \circ a$, at the level $h$ is reached. This results in reward at node $s_{i,h-1}$ at level $h-1$ being equal to $\mathcal{R}(s_{i,h-1}, a) + \gamma\tilde{v}^{(h)}(s'_h)$. As noted before, $\mathcal{R}(s,a)$ is an independent, bounded random variable while $\tilde{v}^{(h)}(\cdot)$ is effectively collected by following a

path all the way to the leaf level. Inductively, we assume that $\tilde{v}^{(h)}(\cdot)$ satisfies the convergence and concentration property for each node or state at level $h$, with $\alpha^{(h)}, \xi^{(h)}, \eta^{(h)}$ as defined per relationship (2.5) - (2.7) and with appropriately defined large enough constant $\beta^{(h)}$. Therefore, by an application of Lemma 4, it follows that this combined reward continues to satisfy (2.12) and (2.13), with $\alpha^{(h)}, \xi^{(h)}, \eta^{(h)}$ as defined per relationship (2.5) - (2.7) and with a large enough constant which we shall denote as $\beta^{(h)}$. These constants are used by the MCTS policy. By an application of Theorem 3, we can obtain the following Lemma 6 regarding the convergence and concentration properties for the reward sequence collected at node $s_{i,h-1}$ at level $h-1$. Similar to the notation in (2.37), let

$$\begin{aligned}
\mu_a^{(h-1)}(s_{i,h-1}) &= \mathbb{E}[\mathcal{R}(s_{i,h-1}, a)] + \gamma \mu_*^{(h)}(s_{i,h-1} \circ a) \\
\mu_*^{(h-1)}(s_{i,h-1}) &= \max_{a \in [K]} \mu_a^{(h-1)}(s_{i,h-1}) \\
a_*^{(h-1)}(s_{i,h-1}) &\in \arg\max_{a \in [K]} \mu_a^{(h-1)}(s_{i,h-1}) \\
\Delta_{\min}^{(h-1)}(s_{i,h-1}) &= \mu_*^{(h-1)}(s_{i,h-1}) - \max_{a \neq a_*^{(h-1)}(s_{i,h-1})} \mu_a^{(h-1)}(s_{i,h-1}).
\end{aligned} \tag{2.38}$$

Again, we shall assume that the maximizer in the set $\arg\max_{a \in [K]} \mu_a^{(h-1)}(s_{i,h-1})$ is unique, i.e. $\Delta_{\min}^{(h-1)}(s_{i,h-1}) > 0$. Define $\tilde{R}_{\max}^{(h-1)} = R_{\max} + \gamma \tilde{R}_{\max}^{(h)}$, where $\tilde{R}^{(H)} = V_{\max} + \varepsilon_0$. Note that all rewards collected at level $h-1$ belong to $[-\tilde{R}_{\max}^{(h-1)}, \tilde{R}_{\max}^{(h-1)}]$.

**Lemma 6.** *Consider a node corresponding to state $s_{i,h-1}$ at level $h-1$ within the MCTS for $i \in [n_{h-1}]$. Let $\tilde{v}^{(h-1)}(s_{i,h-1})_n$ be the total discounted reward collected at $s_{i,h-1}$ during $n \geq 1$ visits. Then, for the choice of appropriately large $\beta^{(h)} > 0$, for a given $\xi^{(h)} > 0$, $\eta^{(h)} \in [\frac{1}{2}, 1)$ and $\alpha^{(h)} > 2$, we have*

*A. Convergence:*

$$\begin{aligned}
&\left| \mathbb{E}\Big[ \frac{1}{n} \tilde{v}^{(h-1)}(s_{i,h-1})_n \Big] - \mu_*^{(h-1)}(s_{i,h-1}) \right| \\
&\leq \frac{2\tilde{R}_{\max}^{(h-1)}(K-1) \cdot \Big( \big( \frac{2(\beta^{(h)})^{\frac{1}{\xi^{(h)}}}}{\Delta_{\min}^{(h-1)}(s_{i,h-1})} \big)^{\frac{1}{1-\eta^{(h)}}} \cdot n^{\frac{\alpha^{(h)}}{\xi^{(h)}(1-\eta^{(h)})}} + \frac{2}{\alpha^{(h)}-2} + 1 \Big)}{n}.
\end{aligned}$$

*B. Concentration: there exist constants, $\beta' > 1$ and $\xi' > 0$ and $1/2 \leq \eta' < 1$ such that for $n \geq 1$, $z \geq 1$,*

$$\begin{aligned}
\mathbb{P}\big( \tilde{v}^{(h-1)}(s_{i,h-1})_n - n\mu_*^{(h-1)}(s_{i,h-1}) \geq n^{\eta'} z \big) &\leq \frac{\beta'}{z^{\xi'}}, \\
\mathbb{P}\big( \tilde{v}^{(h-1)}(s_{i,h-1})_n - n\mu_*^{(h-1)}(s_{i,h-1}) \leq -n^{\eta'} z \big) &\leq \frac{\beta'}{z^{\xi'}},
\end{aligned}$$

*where* $\eta' = \frac{\alpha^{(h)}}{\xi^{(h)}(1-\eta^{(h)})}$, $\xi' = \alpha^{(h)} - 1$, *and* $\beta'$ *is a large enough constant that is function of parameters* $\alpha^{(h)}, \beta^{(h)}, \xi^{(h)}, \eta^{(h)}, \tilde{R}_{\max}^{(h-1)}, K, \Delta_{\min}^{(h-1)}(s_{i,h-1})$.

As before, let us define $\Delta_{\min}^{(h-1)} = \min_{i\in[n_{h-1}]} \Delta_{\min}^{(h-1)}(s_{i,h-1})$. Similarly, we can show that for every node $s_{i,h-1}$, $i \in [n_{h-1}]$, the rate of convergence in Lemma 6 can be uniformly simplified as

$$\begin{aligned}\delta_n^{(h-1)} &= \frac{2\tilde{R}_{\max}^{(h-1)}(K-1)\cdot\Big(\big(\frac{2(\beta^{(h)})^{\frac{1}{\xi^{(h)}}}}{\Delta_{\min}^{(h-1)}}\big)^{\frac{1}{1-\eta^{(h)}}}\cdot n^{\frac{\alpha^{(h)}}{\xi^{(h)}(1-\eta^{(h)})}} + \frac{2}{\alpha^{(h)}-2} + 1\Big)}{n} \\ &= \Theta\Big(n^{\frac{\alpha^{(h)}}{\xi^{(h)}(1-\eta^{(h)})}-1}\Big) \;=\; O\big(n^{\eta-1}\big),\end{aligned}$$

where the last equality holds as $\alpha^{(h)} = \xi^{(h)}(1-\eta^{(h)})\eta^{(h)}$ and $\eta^{(h)} = \eta$. Again, it is worth remarking, inductively, that $\mu_*^{(h-1)}(s_{i,h-1})$ is precisely the value function estimation for $s_{i,h-1}$ at the end of $H-h+1$ steps of value iteration starting with $\hat{V}$.

**Remark.** *(Recursive Relation among Parameters)* With the above development, we are ready to elaborate our choice of parameters in Theorem 1, defined recursively via (2.5)-(2.7). In essence, those parameter requirements originate from our analysis of the non-stationary MAB, i.e., Theorem 3. Recall that from our previous analysis, the key to establish the MCTS guarantee is to recursively argue the convergence and the polynomial concentration properties at each level; that is, we recursively solve the non-stationary MAB problem at each level. In order to do so, we apply our result on the non-stationary MAB (Theorem 3) recursively at each level. Importantly, recall that Theorem 3 only holds when $\xi\eta(1-\eta) \le \alpha < \xi(1-\eta)$ and $\alpha > 2$, under which it leads to the recursive conclusions $\eta' = \frac{\alpha}{\xi(1-\eta)}$ and $\xi' = \alpha - 1$. Using our notation with superscript indicating the levels, this means that apart from the parameters at the leaf level (level $H$) which could be freely chosen, we must choose parameters of other levels recursively so that the following conditions hold:

$$\begin{aligned}&\alpha^{(h)} > 2, \quad \xi^{(h)}\eta^{(h)}(1-\eta^{(h)}) \le \alpha^{(h)} < \xi^{(h)}(1-\eta^{(h)}), \\ &\xi^{(h)} = \alpha^{(h+1)} - 1 \text{ and } \eta^{(h)} = \frac{\alpha^{(h+1)}}{\xi^{(h+1)}(1-\eta^{(h+1)})}.\end{aligned}$$

It is not hard to see that the conditions in Theorem 1 guarantee the above. There might be other sequences of parameters satisfying the requirements, but our particular choice gives cleaner analysis as presented in this chapter.

### ■ 2.7.4 Error Analysis for Value Function Iteration

We now move to the second part of the proof. The value function iteration improves the estimation of optimal value function by iterating the Bellman equation. In effect, the MCTS tree is "unrolling" $H$ steps of such an iteration. Precisely, let $V^{(h)}(\cdot)$ denote the value function after $h$ iterations starting with $V^{(0)} = \hat{V}$. By definition, for any $h \geq 0$ and $s \in \mathcal{S}$,

$$V^{(h+1)}(s) = \max_{a \in [K]} \Big( \mathbb{E}[\mathcal{R}(s,a)] + \gamma V^{(h)}(s \circ a) \Big). \tag{2.39}$$

Recall that value iteration is contractive with respect to $\|\cdot\|_\infty$ norm ([23]). That is, for any $h \geq 0$,

$$\|V^{(h+1)} - V^*\|_\infty \leq \gamma \|V^{(h)} - V^*\|_\infty. \tag{2.40}$$

As remarked earlier, $\mu_*^{(h-1)}(s_{i,h-1})$, the mean reward collected at node $s_{i,h-1}$ for $i \in [n_{h-1}]$ for any $h \geq 1$, is precisely $V^{(H-h+1)}(s_{i,h-1})$ starting with $V^{(0)} = \hat{V}$, the input to MCTS oracle. Therefore, the mean reward collected at root node $s^{(0)}$ of the MCTS tree satisfies $\mu_*^{(0)}(s^{(0)}) = V^{(H)}(s^{(0)})$. Using (2.40), we obtain the following Lemma.

**Lemma 7.** *The mean reward collected under the MCTS policy at root note $s^{(0)}$, $\mu_*^{(0)}(s^{(0)})$, starting with input value function proxy $\hat{V}$ is such that*

$$|\mu_*^{(0)}(s^{(0)}) - V^*(s^{(0)})| \leq \gamma^H \|\hat{V} - V^*\|_\infty.$$

### ■ 2.7.5 Completing Proof of Theorem 1

In summary, using Lemma 6, we conclude that the recursive relationship going from level $h$ to $h-1$ holds for all $h \geq 1$ with level 0 being the root. At root $s^{(0)}$, the query state that is the input to the MCTS oracle, we have that after $n$ total simulations of MCTS, the empirical average of the rewards over these $n$ trial, $\frac{1}{n}\tilde{v}^{(0)}(s_0)_n$ is such that (using the fact that $\alpha^{(0)} = \xi^{(0)}(1-\eta^{(0)})\eta^{(0)}$)

$$\left| \mathbb{E}\Big[\frac{1}{n}\tilde{v}^{(0)}(s_0)_n\Big] - \mu_*^{(0)} \right| = O\Big(n^{\frac{\alpha^{(0)}}{\xi^{(0)}(1-\eta^{(0)})}-1}\Big) \;=\; O\Big(n^{\eta-1}\Big), \tag{2.41}$$

where $\mu_*^{(0)}$ is the value function estimation for $s^{(0)}$ after $H$ iterations of value function iteration starting with $\hat{V}$. By Lemma 7, we have

$$|\mu_*^{(0)} - V^*(s^{(0)})| \leq \gamma^H \varepsilon_0, \tag{2.42}$$

since $\varepsilon_0 = \|\hat{V} - V^*\|_\infty$. Combining (2.41) and (2.42),

$$\left|\mathbb{E}\Big[\frac{1}{n}\tilde{v}^{(0)}(s_0)_n\Big] - V^*(s^{(0)})\right| \leq \gamma^H \varepsilon_0 + O\Big(n^{\eta-1}\Big). \tag{2.43}$$

This concludes the proof of Theorem 1. □

## ■ 2.7.6 Proof of Lemma 4

The convergence property, $\lim_{n\to\infty} \mathbb{E}[\bar{Z}_n] = \mu_X + \rho\mu_Y$, follows simply by linearity of expectation. For concentration, consider the following: since $X$s are i.i.d., bounded random variables taking value in $[-B, B]$, by Hoeffding's inequality [76], we have that for $t \geq 0$,

$$\begin{aligned}\mathbb{P}\big(n\bar{X}_n - n\mu_X \geq nt\big) &\leq \exp\Big(-\frac{t^2 n}{2B^2}\Big), \\ \mathbb{P}\big(n\bar{X}_n - n\mu_X \leq -nt\big) &\leq \exp\Big(-\frac{t^2 n}{2B^2}\Big).\end{aligned} \tag{2.44}$$

Therefore,

$$\begin{aligned}\mathbb{P}\Big(n\bar{Z}_n - n(\mu_X + \rho\mu_Y) \geq n^\eta z\Big) &\leq \mathbb{P}\Big(n\bar{X}_n - n\mu_X \geq \frac{n^\eta z}{2}\Big) + \mathbb{P}\Big(n\bar{Y}_n - n\mu_Y \geq \frac{n^\eta z}{2\rho}\Big) \\ &\leq \exp\Big(-\frac{z^2 n^{2\eta-1}}{8B^2}\Big) + \frac{\beta 2^\xi \rho^\xi}{z^\xi} \\ &\leq \frac{\beta'}{z^\xi},\end{aligned} \tag{2.45}$$

where $\beta'$ is a large enough constant depending upon $\rho, \xi, \beta$ and $B$. The other-side of the inequality follows similarly. This completes the proof. □

## ■ 2.8 Proof of Theorem 2

First, we establish a useful property of nearest neighbor supervised learning presented in Section 2.4.2. This is stated in Section 2.8.1. We will use it, along with the guarantees obtained for MCTS in Theorem 1 to establish Theorem 2 in Section 2.8.2. Throughout, we shall assume the setup of Theorem 2.

## ■ 2.8.1 Guarantees for Supervised Learning

Let $\delta \in (0, 1)$ be given. As stated in Section 2.4.2, let $K(\delta, d) = \Theta(\delta^{-d})$ be the collection of balls of radius $\delta$, say $c_i$, $i \in [K(\delta, d)]$, so that they cover $\mathcal{S}$, i.e. $\mathcal{S} \subset \cup_{i\in[K(\varepsilon,d)]} c_i$. Also, by construction, each of these balls have intersection with $\mathcal{S}$ whose volume is at least $C_d\delta^d$. Let $S = \{s_i : i \in [N]\}$ denote $N$ state samples from $\mathcal{S}$ uniformly at random and

independent of each other. For each state $s \in \mathcal{S}$, let $V : \mathcal{S} \to [-V_{\max}, V_{\max}]$ be such that $|\mathbb{E}[V(s)] - V^*(s)| \le \Delta$. Let the nearest neighbor supervised learning described in Section 2.4.2 produce estimate $\hat{V} : \mathcal{S} \to \mathbb{R}$ using labeled data points $(s_i, V(s_i))_{i\in[N]}$. Then, we claim the following guarantee. Proof can be found in Section 2.8.3.

**Lemma 8.** *Under the above described setup, as long as*

$$N \ge 32 \max(1, \delta^{-2} V_{\max}^2) C_d^{-1} \delta^{-d} \log \frac{K(\delta, d)}{\delta},$$

*i.e.,* $N = \Omega(d\delta^{-d-2} \log \delta^{-1})$,

$$\mathbb{E}\big[\sup_{s\in\mathcal{S}} |\hat{V}(s) - V^*(s)|\big] \le \Delta + (\zeta + 1)\delta + \frac{4V_{\max}\delta^2}{K(\delta, d)}. \tag{2.46}$$

## ■ 2.8.2 Establishing Theorem 2

Using Theorem 1 and Lemma 8, we complete the proof of Theorem 2 under appropriate choice of algorithmic parameters. We start by setting some notation.

To that end, the algorithm as described in Section 2.4.1 iterates between MCTS and supervised learning. In particular, let $\ell \ge 1$ denote the iteration index. Let $m_\ell$ be the number of states that are sampled uniformly at random, independently, over $\mathcal{S}$ in this iteration, denoted as $S^{(\ell)} = \{s_i^{(\ell)} : i \in [m_\ell]\}$. Let $V^{(\ell-1)}$ be the input of value function from prior iteration, using which the MCTS algorithm with $n_\ell$ simulations obtains improved estimates of value function for states in $S^{(\ell)}$ denoted as $\hat{V}^{(\ell)}(s_i^{(\ell)})$, $i \in [m_\ell]$. Using $(s_i^{(\ell)}, \hat{V}^{(\ell)}(s_i^{(\ell)}))_{i\in[m_\ell]}$, the nearest neighbor supervised learning as described above with balls of appropriate radius $\delta_\ell \in (0, 1)$ produces estimate $V^{(\ell)}$ for all states in $\mathcal{S}$. Let $\mathcal{F}^{(\ell)}$ denote the smallest $\sigma$-algebra containing all information pertaining to the algorithm (both MCTS and supervised learning). Define the error under MCTS in iteration $\ell$ as

$$\varepsilon_{\text{mcts}}^{(\ell)} = \mathbb{E}\big[\sup_{s\in\mathcal{S}} \big|\mathbb{E}\big[\hat{V}^{(\ell)}(s)\big|\mathcal{F}^{(\ell-1)}\big] - V^*(s)\big|\big], \tag{2.47}$$

and the error for supervised learning in iteration $\ell$ as

$$\theta_{\text{sl}}^{(\ell)} = \sup_{s\in\mathcal{S}} \big|V^{(\ell)}(s) - V^*(s)\big|, \text{ and } \varepsilon_{\text{sl}}^{(\ell)} = \mathbb{E}\big[\theta_{\text{sl}}^{(\ell)}\big]. \tag{2.48}$$

Recall that in the beginning, we set $V^{(0)}(s) = 0$ for all $s \in \mathcal{S}$. Since $V^*(\cdot) \in [-V_{\max}, V_{\max}]$, we have that $\varepsilon_{\text{sl}}^{(0)} \le V_{\max}$. Further, it is easy to see that if the leaf estimates (i.e., the output of the supervised learning from the previous iteration) is bounded in $[-V_{\max}, V_{\max}]$, then the output of the MCTS algorithm is always bounded in $[-V_{\max}, V_{\max}]$. That is, since $V^{(0)}(s) = 0$ and the nearest neighbor supervised learning produces estimate $V^{(\ell)}$ via

simple averaging, inductively, the output of the MCTS algorithm is always bounded in $[-V_{\max}, V_{\max}]$ throughout every iteration.

With the notation as set up above, it follows that for a given $\delta_\ell \in (0,1)$ with $m_\ell$ satisfying condition of Lemma 8, i.e. $m_\ell = \Omega(d\delta_\ell^{-d-2}\log\delta_\ell^{-1})$, and with the nearest neighbor supervised learning using balls of $\delta_\ell$ radius for estimation, we have the following recursion:

$$\varepsilon_{\text{sl}}^{(\ell)} \le \varepsilon_{\text{mcts}}^{(\ell)} + (\zeta+1)\delta_\ell + \frac{4V_{\max}\delta_\ell^2}{K(\delta_\ell, d)} \;\le\; \varepsilon_{\text{mcts}}^{(\ell)} + C'\delta_\ell, \tag{2.49}$$

where $C'$ is a large enough constant, since $\frac{\delta_\ell^2}{K(\delta_\ell,d)} = \Theta(d\delta_\ell^{d+2})$ which is $O(\delta_\ell)$ for all $\delta_\ell \in (0,1)$. By Theorem 1, for iteration $\ell+1$ that uses the output of supervised learning estimate, $V^{(\ell)}$, as the input to the MCTS algorithm, we obtain

$$\left|\mathbb{E}\big[\hat{V}^{(\ell+1)}(s)\big|\mathcal{F}^{(\ell)}\big] - V^*(s)\right| \le \gamma^{H^{(\ell+1)}}\mathbb{E}\big[\theta_{\text{sl}}^{(\ell)}\big|\mathcal{F}^{(\ell)}\big] + O\big(n_{\ell+1}^{\eta-1}\big), \forall s \in \mathcal{S}, \tag{2.50}$$

where $\eta \in [1/2, 1)$ is the constant utilized by MCTS with fixed depth of tree being $H^{(\ell+1)}$. This then implies that

$$\begin{aligned}
\varepsilon_{\text{mcts}}^{(\ell+1)} &= \mathbb{E}\big[\sup_{s\in\mathcal{S}}\left|\mathbb{E}\big[\hat{V}^{(\ell+1)}(s)\big|\mathcal{F}^{(\ell)}\big] - V^*(s)\right|\big] \\
&\le \gamma^{H^{(\ell+1)}}\mathbb{E}\Big[\mathbb{E}\big[\theta_{\text{sl}}^{(\ell)}\big|\mathcal{F}^{(\ell)}\big]\Big] + O\big(n_{\ell+1}^{\eta-1}\big) \\
&\le \gamma^{H^{(\ell+1)}}\Big(\varepsilon_{\text{mcts}}^{(\ell)} + C'\delta_\ell\Big) + O\big(n_{\ell+1}^{\eta-1}\big).
\end{aligned} \tag{2.51}$$

Denote by $\lambda \triangleq (\frac{\varepsilon}{V_{\max}})^{1/L}$. Note that since the final desired error $\varepsilon$ should be less than $V_{\max}$ (otherwise, the problem is trivial by just outputing 0 as the final estimates for all the states), we have $\lambda < 1$. Let us set the algorithmic parameters for MCTS and nearest neighbor supervised learning as follows: for each $\ell \ge 1$,

$$H^{(\ell)} = \lceil \log_\gamma \frac{\lambda}{8} \rceil, \delta_\ell = \frac{3V_{\max}}{4C'}\lambda^\ell, n_\ell = \kappa_l\Big(\frac{8}{V_{\max}\lambda^\ell}\Big)^{\frac{1}{1-\eta}}, \tag{2.52}$$

where $\kappa_l > 0$ is a sufficiently large constant such that $O\big(n_\ell^{\eta-1}\big) = \frac{V_{\max}}{8}\lambda^\ell$. Substituting these values into (2.51) yields

$$\varepsilon_{\text{mcts}}^{(\ell+1)} = \mathbb{E}\Big[\sup_{s\in\mathcal{S}}\left|\mathbb{E}\big[\hat{V}^{(\ell+1)}(s)|\mathcal{F}^{(\ell)}\big] - V^*(s)\right|\Big] \le \frac{\lambda}{8}\varepsilon_{\text{mcts}}^{(\ell)} + \frac{7V_{\max}}{32}\lambda^{\ell+1}.$$

Note that by (2.50) and (2.52), and the fact that $\varepsilon_{\text{sl}}^{(0)} \le V_{\max}$, we have

$$\varepsilon_{\text{mcts}}^{(1)} \le \frac{\lambda}{8}\varepsilon_{\text{sl}}^{(0)} + \frac{\lambda}{8}V_{\max} \le \frac{\lambda}{4}V_{\max}.$$

It then follows inductively that

$$\varepsilon_{\text{mcts}}^{(\ell)} \leq \lambda^{\ell-1}\varepsilon_{\text{mcts}}^{(1)} = \frac{V_{\max}}{4}\lambda^{\ell}.$$

As for the supervised learning oracle, $\forall s \in \mathcal{S}$, (2.49) implies

$$\mathbb{E}\big[\sup_{s\in\mathcal{S}}\big|V^{(\ell)}(s) - V^*(s)\big|\big] \leq \varepsilon_{\text{mcts}}^{(\ell)} + \frac{3V_{\max}}{4}\lambda^{\ell} \leq V_{\max}\lambda^{\ell}.$$

This implies that

$$\mathbb{E}\big[\sup_{s\in\mathcal{S}}\big|V^{(L)}(s) - V^*(s)\big|\big] \leq V_{\max}\lambda^{L} = \varepsilon.$$

We now calculate the sample complexity, i.e., the total number of state transitions required for the algorithm. During the $\ell$-th iteration, each query of MCTS oracle requires $n_\ell$ simulations. Recall that the number of querying MCTS oracle, i.e., the size of training set $\mathcal{S}^{(\ell)}$ for the nearest neighbor supervised learning step, should satisfy $m_\ell = \Omega(d\delta_\ell^{-d-2}\log\delta_\ell^{-1})$ (cf. Lemma 8). From (2.52), we have

$$H^{(\ell)} = c_0' \log\lambda^{-1}, \quad \delta^{(\ell)} = c_1\lambda^{\ell}, \text{and} \quad n_\ell = c_2'\lambda^{-\ell/(1-\eta)},$$

where $c_0', c_1, c_2'$, are constants independent of $\lambda$ and $\ell$. Note that each simulation of MCTS samples $H^{(\ell)}$ state transitions. Hence, the number of state transitions at the $\ell$-th iteration is given by

$$M^{(\ell)} = m_\ell n_\ell H^{(\ell)}.$$

Therefore, the total number of state transitions after $L$ iterations is

$$\sum_{l=1}^{L} M^{(\ell)} = \sum_{\ell=1}^{L} m_\ell \cdot n_\ell \cdot H^{(\ell)} = O\Big(\varepsilon^{-\left(2+1/(1-\eta)+d\right)} \cdot \big(\log\frac{1}{\varepsilon}\big)^5\Big).$$

That is, for optimal choice of $\eta = 1/2$, the total number of state transitions is $O\big(\varepsilon^{-(4+d)} \cdot \big(\log\frac{1}{\varepsilon}\big)^5\big)$. □

## ■ 2.8.3 Proof of Lemma 8

Given $N$ samples $s_i, i \in [N]$ that are sampled independently and uniformly at random over $\mathcal{S}$, and given the fact that each ball $c_i$, $i \in [K(\delta, d)]$ has at least $C_d\delta^d$ volume shared with $\mathcal{S}$, each of the sample falls within a given ball with probability at least $C_d\delta^d$. Let

$N_i,\ i \in [K(\delta, d)]$ denote the number of samples amongst $N$ samples in ball $c_i$.

Now the number of samples falling in any given ball is lower bounded by a Binomial random variable with parameter $N, C_d\delta^d$. By Chernoff bound for Binomial variable with parameter $n, p$, we have that

$$\mathbb{P}(B(n,p) \le np/2) \le \exp\big(-\frac{np}{8}\big).$$

Therefore, with an application of union bound, each ball has at least $0.5C_d\delta^d N$ samples with probability at least $1 - K(\delta, d)\exp\big(-C_d\delta^d N/8\big)$. That is, for

$$N = 32\max(1, \delta^{-2}V_{\max}^2)C_d^{-1}\delta^{-d}[\log(K(\delta, d) + \log\delta^{-1}],$$

each ball has at least $\Gamma = 16\max(1, \delta^{-2}V_{\max}^2)(\log K(\delta, d) + \log\delta^{-1})$ samples with probability at least $1 - \frac{\delta^2}{K(\delta,d)}$. Define event

$$\mathcal{E}_1 = \{N_i \ge 16\max(1, \delta^{-2}V_{\max}^2)(\log K(\delta, d) + \log\delta^{-1}),\ \forall\ i \in [K(\delta, d)]\}.$$

Then

$$\mathbb{P}(\mathcal{E}_1^c) \le \frac{\delta^2}{K(\delta, d)}.$$

Now, for any $s \in \mathcal{S}$, the nearest neighbor supervised learning described in Section 2.4.2 produces estimate $\hat{V}(s)$ equal to the average value of observations for samples falling in ball $c_{j(s)}$. Let $N_{j(s)}$ denote the number of samples in ball $c_{j(s)}$. To that end,

$$\begin{aligned}
\big|\hat{V}(s) - V^*(s)\big| &= \bigg|\frac{1}{N_{j(s)}}\Big(\sum_{i:s_i\in c_{j(s)}} V(s_i) - V^*(s)\Big)\bigg| \\
&= \bigg|\frac{1}{N_{j(s)}}\Big(\sum_{i:s_i\in c_{j(s)}} V(s_i) - \mathbb{E}[V(s_i)]\Big)\bigg| + \bigg|\frac{1}{N_{j(s)}}\Big(\sum_{i:s_i\in c_{j(s)}} \mathbb{E}[V(s_i)] - V^*(s_i)\Big)\bigg| \\
&\quad + \bigg|\frac{1}{N_{j(s)}}\Big(\sum_{i:s_i\in c_{j(s)}} V^*(s_i) - V^*(s)\Big)\bigg|.
\end{aligned}$$

For the first term, since for each $s_i \in c_{j(s)}$, $V(s_i)$ is produced using independent randomness via MCTS, and since the output $V(s_i)$ is a bounded random variable, using Hoeffding's

inequality, it follows that

$$\mathbb{P}\Big(\Big|\frac{1}{N_{j(s)}}\Big(\sum_{i:s_i\in c_{j(s)}} V(s_i) - \mathbb{E}[V(s_i)]\Big)\Big| \geq \Delta_1\Big) \leq 2\exp\Big(-\frac{N_{j(s)}\Delta_1^2}{8V_{\max}^2}\Big).$$

The second term is no more than $\Delta$ due to the guarantee given by MCTS as assumed in the setup. And finally, the third term is no more than $\zeta\delta$ due to Lipschitzness of $V^*$. To summarize, with probability at least $1 - 2\exp\Big(-\frac{N_{j(s)}\Delta_1^2}{8V_{\max}^2}\Big)$, we have that

$$\Big|\hat{V}(s) - V^*(s)\Big| \leq \Delta_1 + \Delta + \zeta\delta.$$

As can be noticed, the algorithm produces the same estimate for all $s \in \mathcal{S}$ such that they map to the same ball. And there are $K(\delta, d)$ such balls. Therefore, using union bound, it follows that with probability at least $1 - 2K(\delta,d)\exp\Big(-\frac{(\min_{i\in[K(\delta,d)]} N_i)\Delta_1^2}{8V_{\max}^2}\Big)$,

$$\sup_{s\in\mathcal{S}}\Big|\hat{V}(s) - V^*(s)\Big| \leq \Delta_1 + \Delta + \zeta\delta.$$

Under event $\mathcal{E}_1$, $\min_{i\in[K(\delta,d)]} N_i \geq 16\max(1, \delta^{-2}V_{\max}^2)(\log K(\delta,d) + \log\delta^{-1})$. Therefore, under event $\mathcal{E}_1$, by choosing $\Delta_1 = \delta$, we have

$$\sup_{s\in\mathcal{S}}\Big|\hat{V}(s) - V^*(s)\Big| \leq \Delta + (\zeta+1)\delta,$$

with probability at least $1 - \frac{2\delta^2}{K(\delta,d)}$. When event $\mathcal{E}_1$ does not hold or the above does not hold, we have trivial error bound of $2V_{\max}$ on the error. Therefore, we conclude that

$$\mathbb{E}\Big[\sup_{s\in\mathcal{S}}\Big|\hat{V}(s) - V^*(s)\Big|\Big] \leq \Delta + (\zeta+1)\delta + \frac{4V_{\max}\delta^2}{K(\delta,d)}.$$

□

## ■ 2.9 Extension of Theorem 1 for Stochastic Environment

In the previous sections, we have successfully analyzed the non-asymptotic behavior of MCTS under deterministic transitions and studied the complexity of the resulted reinforcement learning algorithm when combined with nearest neighbor supervised learning. In the last part of this chapter, we show that our analysis can be naturally extended to handle the stochastic environment in essentially the same manner.

We have established Theorem 1 when the transition kernel is deterministic. We now explain how to extend the results to the setting with stochastic transition kernel. We do so

by effectively mapping the stochastic setting to a deterministic setting as discussed next.

We start by defining the stochastic environment. Recall that when an action $a$ is taken at state $s$, the next state is $s'$ with probability $\mathcal{P}(s'|s,a)$. In the deterministic setting, we have $\mathcal{P}(s'|s,a) \in \{0,1\}$, while in the stochastic setting, we allow for $\mathcal{P}(s'|s,a) \in [0,1]$. We further consider the following setup. Let there be a fixed $\phi > 0$ so that

$$\inf\{\mathcal{P}(s'|s,a) : \mathcal{P}(s'|s,a) \neq 0,\ s,s' \in \mathcal{S}, a \in \mathcal{A}\} \geq \phi. \tag{2.53}$$

Let $\text{supp}(s,a)$ be the support of the distribution $\mathcal{P}(\cdot|s,a)$. Due to (2.53), $|\text{supp}(s,a)| \leq \lfloor \frac{1}{\phi} \rfloor \equiv M$. That is, the number of next state reachable for a given state $s$ under an action $a$ is bounded by a constant $M$ for all $s,a$.

Let us consider the MCTS algorithm for such a stochastic setting. At a node (i.e., state) at depth $h$, the action with the highest sum of average reward and a polynomial bonus is selected. A next state at depth $h+1$ is reached, and the process is repeated until a fix depth $H$. We then update the corresponding statistics of the nodes and the selected actions at each depth, and this finishes one iteration of the simulation. Since the transitions are stochastic, for state (node) $s$ at each depth, each action $a \in \mathcal{A}$ would have up to $|\text{supp}(s,a)| \leq M$ children nodes. In contrast, for the deterministic case, each action leads to a unique state at the next depth (as shown in Figure 2.1, where each edge represents an action and connects a node $s$ at depth $h$ to a unique next state $s'$ at depth $h+1$). However, despite of the distinct difference, we can map the stochastic scenario back to the deterministic setting via a simple transformation. Specifically, given the state $s$ at depth $h$ and action $a$, though there are multiple next states, for the purpose of MCTS decision, we assign a "meta-edge" corresponding to each action $a \in \mathcal{A}$ for a given state $s \in \mathcal{S}$. This edge connects $s$ via action $a$ to all of its next states in $\text{supp}(s,a)$. This is illustrated in Figure 2.2, where each thick edge is a "meta-edge" representing an action in $\mathcal{A}$.

In the deterministic setting, at the end of each simulation step, the rewards of nodes and edges were updated along the entire path visited in the simulation step as described in Algorithm 1. In the stochastic setting, we perform the same operation, i.e., updating the rewards for each node (state) and each action (i.e., the meta edge) in the same manner. Note that now we might have a larger tree due to multiple children associated with the same action for a given state. Finally, while similar in spirit, the key difference lies in how we selection an action $a \in \mathcal{A}$ at a given state $s \in \mathcal{S}$ at depth $h$ of the tree in a simulation step. In the deterministic setting, we simply utilize the sum of the empirical average return and the polynomial bonus term associated with the action (or the edge), as described in (2.4). In stochastic setting, for each action $a$ at a state $s$, instead, we use a weighted sum of the empirical average returns associated with all possible next states, with weights simply being

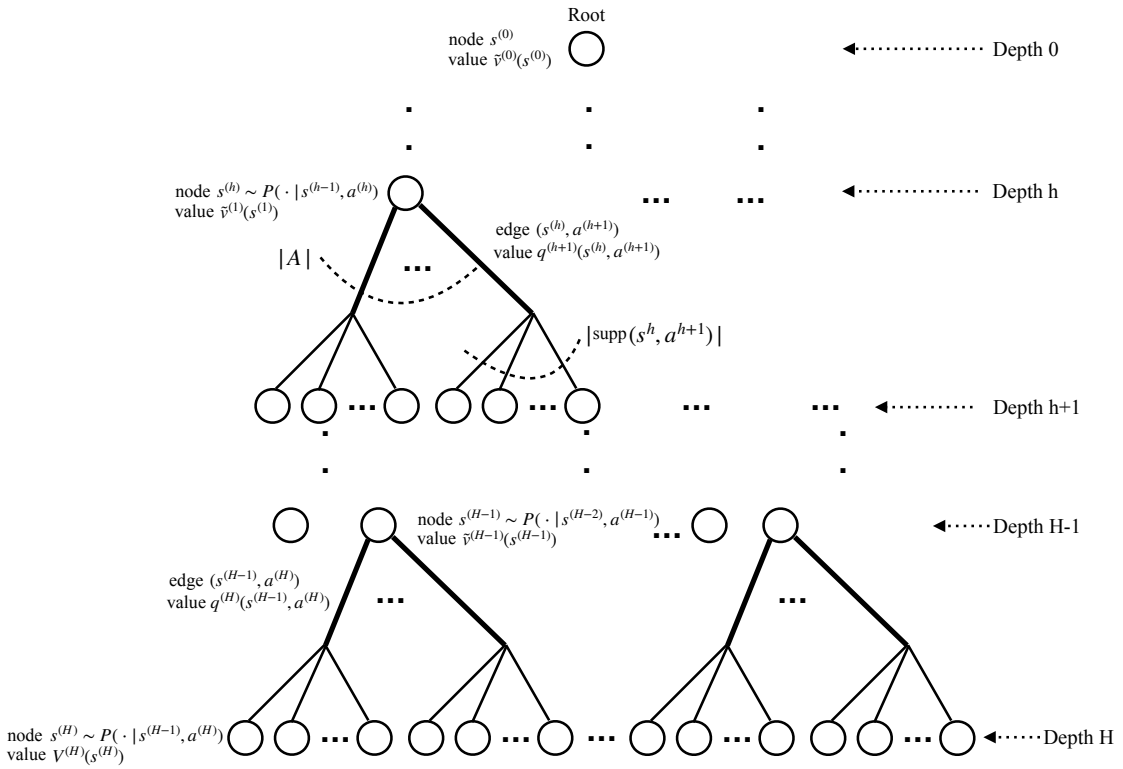



**Figure 2.2.** MCTS with stochastic transitions.

the empirical frequency of visiting each next state in $\mathrm{supp}(s, a)$ so far. We use a similar polynomial bonus term for each action.

With the modifications elaborated above, we can now reuse the majority of our previous analysis. Recall that to establish the desired theorem for MCTS with deterministic transitions, we recursively argue the convergence and polynomial concentration properties at each depth. That is, starting with the convergence and concentration properties for nodes at depth $h + 1$, we show the convergence and concentration properties for nodes at depth $h$; and then recursively apply this process until we reach the root node. More precisely, the induction step is completed by analyzing a non-stationary MAB problem where the (non-stationary) outcomes of each arm converge and polynomially concentrate. In stochastic setting, the algorithm dynamics are almost the same as that for deterministic setting, except that upon taking an arm (action), there is additional randomness determining which children in $\mathrm{supp}(s, a)$ we transition to. Suppose that we can argue that the non-stationary outcomes of each arm, after accounting for the stochastic transition through weighted average with empirical frequency, have the same convergence and polynomially concentration properties as the children nodes. Consequently, we can apply the analysis we developed for the deterministic case, by following the same line of induction argument.

Specifically, we can reduce the analysis of MCTS for stochastic settings to that of the deterministic settings as shown in Figure 2.3. We view the children nodes associated with one action collectively as one "meta-node" corresponding to the action, i.e., the "meta-node" encapsulates the randomness of the transitions and the non-stationary reward processes at the children nodes. At depth $h+1$, starting with the convergence and concentration properties for the non-stationary reward processes at each child node, we show that the reward process at the "meta-node" has the same convergence and concentration properties. The action selection problem at each node/state for the stochastic setting then is reduced to the MAB problem we have analyzed in the deterministic setting, for which we have established the convergence and concentration properties for the parent nodes at depth $h$. By following the proof for the deterministic settings, we shall obtain the guarantees for MCTS with stochastic environments. To summarize, it is clear that to establish the desired results for MCTS, we only need to fill in the missing step of arguing the convergence and concentration properties of the "meta-node"; the rest of the proof then exactly follows without modifications.

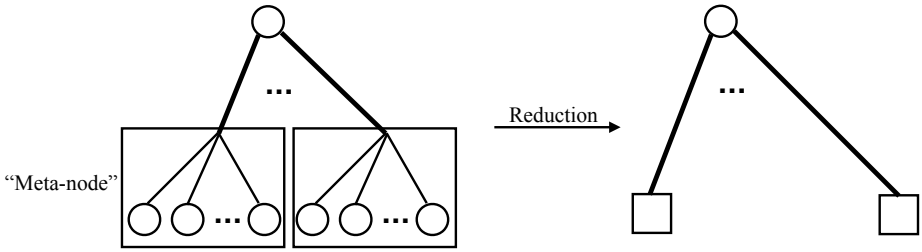



**Figure 2.3.** Reduce the stochastic transitions to a single "meta-node" for each action.

To this end, we consider a mathematical formulation that precisely describes the action selection problem at a node with stochastic transition. Consider a multinomial distribution over $[M] = \{1, \ldots, M\}$ with $p_m \geq \phi$ being probability of observing outcome $m \in [M]$. We denote the distribution by $\text{Dist}(p)$. Let us consider a sequence of i.i.d. random variables $\{Y_i, i \in \mathbb{N}^+\}$, where $Y_i \sim \text{Dist}(p)$. Consider $M$ random processes (possibly dependent) $\{X_{m,t}, t \in \mathbb{N}^+\}$ for $1 \leq m \leq M$. Define a random process $\{Z_i, i \in \mathbb{N}^+\}$ as follows: $Z_i = \sum_{m=1}^{M} \mathbb{I}\{Y_i = m\} X_{m,N(m,i-1)+1}$, where $N(m, i-1) = \sum_{j=1}^{i-1} \mathbb{I}\{Y_j = m\}$ is the total number of times that the $m$th outcome has been generated up to (and including) time $i-1$. In the context of MAB with stochastic transition, the introduced random processes are associated with one arm $a$ as follows: playing action $a$ leads to a random next state in $[M]$ according to $\text{Dist}(p)$; state $m \in [M]$ is associated with a reward sequence $\{X_{m,t}, t \in \mathbb{N}^+\}$; $Z_i$ represents the reward obtained by playing the action $a$ for the $i$th time. We establish that if for each $m \in [M]$, the random process $\{X_{m,t}, t \in \mathbb{N}^+\}$ satisfies the convergence and the polynomial

concentration properties, then so does the random process $\{Z_i\}$, as stated in the following lemma.

**Lemma 9.** *Suppose that the $M$ random processes $\{X_{m,t}, t \in \mathbb{N}^+\}$, $1 \le m \le M$, satisfy*

*A. Convergence: for $n \ge 1$, with notation $\bar{X}_{m,n} = \frac{1}{n}\big(\sum_{t=1}^{n} X_{m,t}\big)$,*

$$\lim_{n\to\infty} \mathbb{E}[\bar{X}_{m,n}] = \mu_m, \quad \forall\, 1 \le m \le M.$$

*B. Concentration: there exist constants, $\beta > 1$, $\xi > 1$, $1/2 \le \eta < 1$ such that for $n \ge 1$ and $z \ge 1$,*

$$\mathbb{P}\big(n\bar{X}_{m,n} - n\mu_m \ge n^{\eta} z\big) \le \frac{\beta}{z^{\xi}}, \quad \mathbb{P}\big(n\bar{X}_{m,n} - n\mu_m \le -n^{\eta} z\big) \le \frac{\beta}{z^{\xi}}, \quad \forall\, 1 \le m \le M.$$

*Then, the random process $\{Z_i, i \in \mathbb{N}^+\}$ satisfies*

*A. Convergence: for $n \ge 1$, with notation $\bar{Z}_n = \frac{1}{n}\big(\sum_{i=1}^{n} Z_i\big)$,*

$$\lim_{n\to\infty} \mathbb{E}[\bar{Z}_n] = \sum_{m=1}^{M} p_m \mu_m.$$

*B. Concentration: there exist constant $\beta' > 1$ depending upon $M, \xi, \beta$ such that for $n \ge 1$ and $z \ge 1$,*

$$\mathbb{P}\Big(n\bar{Z}_n - n\big(\sum_{m=1}^{M} p_m\mu_m\big) \ge n^{\eta} z\Big) \le \frac{\beta'}{z^{\xi}}, \quad \mathbb{P}\Big(n\bar{Z}_n - n\big(\sum_{m=1}^{M} p_m\mu_m\big) \le -n^{\eta} z\Big) \le \frac{\beta'}{z^{\xi}}.$$

As discussed, with Lemma 9, the proof in the previous sections is then readily extended to the stochastic setting. One important aspect that is worth mentioning is that the constants related to the polynomial rate, $\eta$ and $\xi$, are preserved and remain unchanged from the processes $\{X_{m,\cdot}\}$ to the process $Z$, i.e., the "meta-nodes" has the same polynomial rate as the children nodes. Only the constant $\beta$ is different. This means that the proof of Theorem 1 can be applied with a simple change of a different constant $\beta'$. Particularly, Theorem 1 holds with the same rate of convergence, i.e., $O(n^{\eta-1})$. Finally, one may notice that in Lemma 9, for the concentration of $\{X_{m,t}, t \in \mathbb{N}^+\}$, we assume $\xi > 1$ instead of a more general choice $\xi > 0$ (cf., Section 2.5). This is indeed not an issue, as one can easily verify that the conditions in Theorem 1, i.e., choosing a large $\xi^{(H)}$ at depth $H$ and using the algorithmic choices (2.5) - (2.7), implicitly guarantees $\xi > 1$ for every depth recursively.

### ■ 2.9.1 Proof of Lemma 9

Fix $n$. Note that according to the generating process, we can re-write $\bar{Z}_n$ as

$$\bar{Z}_n = \frac{1}{n}\Big(\sum_{m=1}^{M}\sum_{i=1}^{\mathcal{N}_m} X_{m,i}\Big),$$

where $\mathcal{N}_m$, $1 \le m \le M$ are random variables such that $\sum_{m=1}^{M}\mathcal{N}_m = n$ and the marginal distribution of $\mathcal{N}_m \sim \text{Binomial}(n, p_m)$, i.e., $\mathcal{N}_m$ is the number of times the $m$th outcome is generated according to the distribution $\text{Dist}(p)$ after $n$ trials. By Hoeffding's inequality, we have that for $1 \le m \le M$ and $\delta \ge 0$,

$$\mathbb{P}(\mathcal{N}_m\mu_m - np_m\mu_m \ge \delta) \le \exp\Big(-\frac{2\delta^2}{n\mu_m^2}\Big).$$

Therefore,

$$\mathbb{P}(\mathcal{N}_m\mu_m - np_m\mu_m \ge p_m n^{\eta} z) \le \exp\Big(-\frac{2p_m^2 z^2 n^{2\eta-1}}{\mu_m^2}\Big) \le \frac{\beta_m}{z^{\xi}},$$

where $\beta_m$ is a large enough constant depending upon $\xi, p_m$ and $\mu_m$ and importantly, independent of $n$. Note that the last step follows because $\eta \ge 1/2$ and the exponential tail resulted from the Hoeffding's inequality decays faster than a polynomial one. We have that

$$\begin{aligned}
&\mathbb{P}\Big(n\bar{Z}_n - \sum_{m=1}^{M} np_m\mu_m \ge n^{\eta} z\Big) \\
&\le \mathbb{P}\Big(n\bar{Z}_n - \sum_{m=1}^{M} np_m\mu_m \ge \sum_{m=1}^{M}\frac{\mathcal{N}_m^{\eta} z}{2M} + \sum_{m=1}^{M}\frac{p_m n^{\eta} z}{2}\Big) \qquad (2.54)\\
&= \mathbb{P}\Big(\sum_{m=1}^{M}\mathcal{N}_m\bar{X}_{m,\mathcal{N}_m} - \sum_{m=1}^{M} np_m\mu_m \ge \sum_{m=1}^{M}\frac{\mathcal{N}_m^{\eta} z}{2M} + \sum_{m=1}^{M}\frac{p_m n^{\eta} z}{2}\Big) \\
&\le \sum_{m=1}^{M}\mathbb{P}\Big(\mathcal{N}_m\bar{X}_{m,\mathcal{N}_m} - np_m\mu_m \ge \frac{\mathcal{N}_m^{\eta} z}{2M} + \frac{p_m n^{\eta} z}{2}\Big). \qquad (2.55)
\end{aligned}$$

Note that (2.54) follows because the following holds almost surely:

$$\sum_{m=1}^{M}\frac{\mathcal{N}_m^{\eta} z}{2M} + \sum_{m=1}^{M}\frac{p_m n^{\eta} z}{2} \le \sum_{m=1}^{M}\frac{n^{\eta} z}{2M} + \sum_{m=1}^{M}\frac{p_m n^{\eta} z}{2} = n^{\eta} z.$$

Further, (2.55) holds due to the fact that

$$\mathbb{P}(A + B \ge C + D) \le \mathbb{P}(A \ge C \text{ or } B \ge D) \le \mathbb{P}(A \ge C) + \mathbb{P}(B \ge D).$$

To continue, we have that

$$\begin{aligned}
&\mathbb{P}\Big(\mathcal{N}_m\bar{X}_{m,\mathcal{N}_m} - np_m\mu_m \geq \frac{\mathcal{N}_m^\eta z}{2M} + \frac{p_m n^\eta z}{2}\Big) \\
&= \mathbb{P}\Big(\mathcal{N}_m\bar{X}_{m,\mathcal{N}_m} - \mathcal{N}_m\mu_m + \mathcal{N}_m\mu_m - np_m\mu_m \geq \frac{\mathcal{N}_m^\eta z}{2M} + \frac{p_m n^\eta z}{2}\Big) \\
&\leq \mathbb{P}\Big(\mathcal{N}_m\bar{X}_{m,\mathcal{N}_m} - \mathcal{N}_m\mu_m \geq \frac{\mathcal{N}_m^\eta z}{2M}\Big) + \mathbb{P}\Big(\mathcal{N}_m\mu_m - np_m\mu_m \geq \frac{p_m n^\eta z}{2}\Big) \\
&= \mathbb{E}\Big[\mathbb{P}\Big(\mathcal{N}_m\bar{X}_{m,\mathcal{N}_m} - \mathcal{N}_m\mu_m \geq \frac{\mathcal{N}_m^\eta z}{2M} \mid \mathcal{N}_m\Big)\Big] + \mathbb{P}\Big(\mathcal{N}_m\mu_m - np_m\mu_m \geq \frac{p_m n^\eta z}{2}\Big) \\
&\leq \mathbb{E}\Big[\frac{\beta(2M)^\xi}{z^\xi}\Big] + \frac{2^\xi\beta_m}{z^\xi} \\
&\leq \frac{\beta'_m}{z^\xi}, \qquad (2.56)
\end{aligned}$$

where $\beta'_m = \beta(2M)^\xi + 2^\xi\beta_m$. Note that $\mathbb{P}\Big(\mathcal{N}_m\bar{X}_{m,\mathcal{N}_m} - \mathcal{N}_m\mu_m \geq \frac{\mathcal{N}_m^\eta z}{2M} \mid \mathcal{N}_m\Big) \leq \frac{\beta(2M)^\xi}{z^\xi}$ holds, because if $z \geq 2M$, the concentration inequality for $\{\bar{X}_{m,\cdot}\}$ assumed in the lemma applies; and if $1 \leq z < 2M$, the R.H.S. of the above inequality is larger than 1 since $\beta > 1$ and the inequality trivially holds. Combining (2.55) and (2.56), we have that

$$\mathbb{P}\Big(n\bar{Z}_n - \sum_{m=1}^M np_m\mu_m \geq n^\eta z\Big) \leq \sum_{m=1}^M \frac{\beta'_m}{z^\xi} \leq \frac{\beta'}{z^\xi},$$

where $\beta' = M\max_{1\leq m\leq M}\beta'_m$. The other side of the inequality follows similarly, and this completes the proof of the desired concentration property of $\bar{Z}_n$.

For convergence, note that we have established the concentration property that for $z \geq 1$:

$$\mathbb{P}\Big(|\bar{Z}_n - \sum_{m=1}^M p_m\mu_m| \geq n^{\eta-1}z\Big) \leq \frac{2\beta'}{z^\xi}.$$

Therefore,

$$\begin{aligned}
\mathbb{E}\Big[|\bar{Z}_n - \sum_{m=1}^M p_m\mu_m|\Big] &= \int_0^\infty \mathbb{P}\Big(|\bar{Z}_n - \sum_{m=1}^M p_m\mu_m| \geq s\Big)ds \\
&= \int_0^{n^{\eta-1}} \mathbb{P}\Big(|\bar{Z}_n - \sum_{m=1}^M p_m\mu_m| \geq s\Big)ds + \int_{n^{\eta-1}}^\infty \mathbb{P}\Big(|\bar{Z}_n - \sum_{m=1}^M p_m\mu_m| \geq s\Big)ds \\
&\leq n^{\eta-1} + \int_{n^{\eta-1}}^\infty \frac{2\beta' n^{\xi(\eta-1)}}{s^\xi}ds \\
&= n^{\eta-1} + \frac{2\beta' n^{\eta-1}}{\xi-1},
\end{aligned}$$

where the integral is finite because $\xi > 1$ by assumption in the lemma. Therefore,

$$\lim_{n\to\infty} \left|\mathbb{E}\Big[\bar{Z}_n - \sum_{m=1}^{M} p_m \mu_m\Big]\right| \leq \lim_{n\to\infty} \mathbb{E}\Big[\Big|\bar{Z}_n - \sum_{m=1}^{M} p_m \mu_m\Big|\Big] \leq \lim_{n\to\infty} \Big(n^{\eta-1} + \frac{2\beta' n^{\eta-1}}{\xi - 1}\Big) = 0.$$

The limit is 0 because $1/2 \leq \eta < 1$. The above implies that $\lim_{n\to\infty} \mathbb{E}[\bar{Z}_n] = \sum_{m=1}^{M} p_m \mu_m$, which establishes the desired convergencd property of $\bar{Z}_n$. This completes the proof of Lemma 9. □

## ■ 2.10 Chapter Summary

This chapter marks the beginning of our investigation on data efficiency of reinforcement learning. In particular, by taking inspiration from the empirical successes, we are interested in understanding the complexity of Monte Carlo Tree Search (MCTS) and its usage in recent RL algorithms. To this end, this chapter introduces a *correction* of the popular MCTS approach for improved value function estimation for a given state, using an existing value function estimate for the entire state space. This correction was obtained through careful, rigorous analysis of a non-stationary Multi-arm Bandit (MAB) where rewards are dependent and non-stationary. In particular, we analyzed a variant of the classical Upper Confidence Bound policy for such a MAB. Using this as a building block, we establish rigorous performance guarantees for the *corrected* version of MCTS proposed in this chapter. This, to the best of our knowledge, is the first mathematically correct analysis of the UCT policy despite its popularity since it has been proposed in literature [94, 95]. We further establish that the proposed MCTS method, when combined with nearest neighbor supervised learning, leads to near optimal sample complexity for obtaining estimation of value function within a given tolerance, where the optimality is in the minimax sense. This suggests the tightness of our analysis as well as the utility of the MCTS approach.

We take a note that much of this work was inspired by the success of AlphaGo Zero (AGZ) which utilizes MCTS combined with supervised learning. Interestingly enough, the correction of MCTS suggested by our analysis is qualitatively similar to the version of MCTS utilized by AGZ as reported in practice. This seeming coincidence may suggest further avenue for practical utility of versions of the MCTS proposed in this chapter and is an interesting direction for future work.

Finally, we remark that the focus of this chapter is on the theoretical understanding of MCTS which we believe is important but currently lacking. One important question that is not fully addressed, however, is why MCTS, when combined with supervised learning, performs so well in practice, as evident by AGZ. We provide an analysis on the data efficiency of such a simplified algorithm with nearest neighbor regression, which leads to a sample

complexity of $\tilde{O}(\varepsilon^{-(4+d)})$. One may wonder if this can be improved or in general, how MCTS can be better utilized in practice. We note that ignoring many ad-hoc modifications, there is one important "trick" in practice: re-using simulation data. If only used in its vanilla format, we tend to not expect MCTS to perform significantly better in terms of theoretical (worst-case) sample complexity. To obtain a good estimate, each query of the MCTS oracle necessarily incurs a large number of samples, which are then discarded after that query. This leads to a significant waste of samples. However, it is not what practitioners might do. For example, in AGZ, "the search tree is reused at subsequent time steps: the child node corresponding to the played action becomes the new root node; the subtree below this child is retained along with all its statistics, while the remainder of the tree is discarded." Practical schemes that re-use samples could greatly improve the sample complexity and real performance. However, they necessarily create complicated, dependent random variables that are harder to analyze theoretically and are beyond the scope of this chapter. Nevertheless, in Chapter 4 where we design efficient stable policy, we will give an example of a naive data re-use scheme and show how re-use data can provably improve the data efficiency per query for a simpler Monte Carlo oracle. While the treatment in Chapter 4 is certainly simplified, it offers preliminary evidence on the importance of such a trick. Analyzing data reuse in its full generality and provably demonstrate its value is undoubtedly an important open question in explaining the success of recent MCTS-related RL algorithms. We hope that this chapter could serve as an attempt to provide some foundational understanding on MCTS as well as the combination of MCTS and supervised learning, and motivate further studies on the related open problems.

Chapter 3

# Data-efficient "Low-rank" RL

In this chapter, we start our journey on designing data-efficient reinforcement learning algorithms, focusing on compact domains first. In particular, we consider the question of learning an $\varepsilon$-optimal $Q$-function under continuous state and action spaces. As previewed in Chapter 1, if $Q$-function is Lipschitz continuous, then the minimal sample complexity for estimating $\varepsilon$-optimal $Q$-function is known to scale as $\Omega(\frac{1}{\varepsilon^{d_1+d_2+2}})$ per classical non-parametric learning theory, where $d_1$ and $d_2$ denote the dimensions of the state and action spaces respectively. To overcome the barrier in sample complexity, our methodology is to first take a spectrum viewpoint of the the $Q$-function: when viewed as a kernel, the $Q$-function induces a Hilbert-Schmidt operator and hence possesses square-summable spectrum. This spectral property motivates us to consider a parametric class of $Q$-functions parameterized by its "rank" $r$, which contains all Lipschitz $Q$-functions as $r \to \infty$. As the key contribution of this chapter, we develop a simple, iterative learning algorithm that finds $\varepsilon$-optimal $Q$-function with sample complexity of $\widetilde{O}(\frac{1}{\varepsilon^{\max(d_1,d_2)+2}})$ when the optimal $Q$-function has low rank $r$ and the discounting factor $\gamma$ is below a certain threshold. Thus, this provides an exponential improvement in sample complexity. While our initial motivation is for RL problems with continuous domains, which are less studied in literature, our methodology turns out to be generic and equally effective for problems with finite/discrete domains. The following table summarizes the improvement of sample complexity for various settings.

**Table 3.1.** Informal summary of sample complexity results for three different state/action space configurations: our results, a few selected from literature, and the lower bounds.

| Setting | Our Results | Selected from Literature | | Lower Bound |
|---|---|---|---|---|
| Cont. $\mathcal{S}$ & Cont. $\mathcal{A}$ | $\tilde{O}(\frac{1}{\varepsilon^{\max\{d_1,d_2\}+2}})$ | N/A | | $\Omega(\frac{1}{\varepsilon^{d_1+d_2+2}})$ [164] |
| Cont. $\mathcal{S}$ & Finite $\mathcal{A}$ | $\tilde{O}(\frac{1}{\varepsilon^{d_1+2}})$ | $\tilde{O}(\frac{1}{\varepsilon^{d_1+3}})$ [143] | $\tilde{O}(\frac{1}{\varepsilon^{d_1+2}})$ [181] | $\tilde{\Omega}(\frac{1}{\varepsilon^{d_1+2}})$ [143] |
| Finite $\mathcal{S}$ & Finite $\mathcal{A}$ | $\tilde{O}(\frac{\max(\lvert\mathcal{S}\rvert,\lvert\mathcal{A}\rvert)}{\varepsilon^2})$ | $\tilde{O}(\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^3\varepsilon^2})$ [146] | $\tilde{O}(\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^4\varepsilon^2})$ [147] | $\tilde{\Omega}(\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^3\varepsilon^2})$ [15] |

To enable our result, we develop a novel Matrix Estimation (ME) algorithm that faithfully estimates an unknown low-rank matrix in the $\ell_\infty$ sense even in the presence of arbitrary,

bounded noise, which might be of interest in its own right. Such an ME algorithm is absent in the literature but is crucial for the success of our efficient algorithm: $Q$-value estimates for only a limited number of state-action pairs will be generalized to a much larger state-action set via the ME algorithm. This is as if we had explored the full space, but indeed achieved with significantly fewer samples. As a preview, Table 3.2 compares our method with some existing ME literature to highlight the difficulties of the setting considered and our contributions.

**Table 3.2.** Comparison of different ME methods with different guarantees. Ours is the only method that provides entry-wise guarantee while allowing for arbitrary, bounded error in each entry.

| Method | Noise Model | Error Guarantees | Sampling Model | # of Samples |
|---|---|---|---|---|
| Our Method | bounded arbitrary | entrywise | adaptive | $O(n)$ |
| Convex Relaxation | noiseless | exact | independent w.p. $p$ | $O(n \log^2 n)$ |
| [32, 30, 97] | bounded arbitrary | Frobenius | independent w.p. $p$ | $O(n \log^2 n)$ |
| Spectral Thresholding [34] | zero-mean | Frobenius | independent w.p. $p$ | $O(n^{1+c})$ |
| Factorization (noncvx) [39] | zero-mean | entrywise | independent w.p. $p$ | $O(n \log^3 n)$ |

Overall, to the best of our knowledge, this is the first work to show a provable, quantitative utility of exploiting the low-rank structure to reduce sample complexity in $Q$-learning for problems with continuous state space and continuous action space.

**Organization of Chapter 3.** Chapter 3 is organized as follows. We begin with reviewing the related literature in Section 3.1 to provide an informed background. In Section 3.2, we introduce a formal representation theorem of $Q^*$ which motivates the study of the low-rank structure. We then propose our efficient RL algorithm using low-rank ME in Section 3.3. The generic convergence and sample complexity results are established in Section 3.4, under a suitable assumption on the ME method. Section 3.5 is dedicated to the development of our new ME method that fulfills the requirement, with Section 3.6 containing the corresponding proofs. To probe the efficacy of our framework, we provide empirical evidence in Section 3.7. In Section 3.8, we offer a further discussion on aspects of our ME method, and then we summarize and conclude this chapter in Section 3.9. For a better readability, a few technical proofs as well as the detailed experimental setup are deferred to Appendix B.

## ■ 3.1 Related Work

We discuss related work on the sample complexity of reinforcement learning and matrix estimation to better contextualize our study.

**Reinforcement Learning.** RL problems with both continuous state and action spaces received significantly less attention in literature. While there are practical RL algorithms

to deal with continuous domains [166, 106, 69, 110], theoretical understanding on this class of problems, especially on sample complexity, is very limited [9]. This motivates the investigation of this chapter on improving efficiency for such scenarios. Since we interpolate our estimates to the entire space via non-parametric regression without making any additional model assumptions, a comparison with the non-parametric minimax rate $\Omega(\frac{1}{\varepsilon^{d_1+d_2+2}})$ for learning Lipschitz function [154, 164] is meaningful.

Our "low-rank" algorithm and proofs are general, which can be reduced to low-rank settings with a finite (discrete) space in a similar manner. We offer a high-level comparison in Table 3.1 with a few selected work to help readers see how our approach fares with others from literature. This is by no means a complete illustration or a strict comparison on sample complexity, given the vast literature on the finite settings and the various problem settings considered. Rather, we intend to convey a rough sense of how our efficient algorithm performs, and especially what we gain in sample complexity with exploiting low-rank structure. In continuous state and action spaces, our algorithm effectively removes the dependence on the smaller dimension by leveraging the low-rank factorization, i.e., sample complexity is improved from $\Omega(\frac{1}{\varepsilon^{d_1+d_2+2}})$ to $\tilde{O}(\frac{1}{\varepsilon^{\max(d_1,d_2)+2}})$. The same heuristic in fact carries over to the finite cases, where the dependence on the size of the smaller space is "removed," i.e., the sample complexity depends on $|\mathcal{S}|$ instead of $|\mathcal{S}||\mathcal{A}|$, assuming $|\mathcal{S}| \geq |\mathcal{A}|$. For problems with continuous state space and finite action space, the lower bound scales as $\tilde{\Omega}(\frac{1}{\varepsilon^{d+2}})$ [143]. In [143], authors consider learning the $Q$-function in a single sample path with non-parametric regression methods, whereas [181] considers learning the $Q$-function with sparse neural networks when re-sampling i.i.d. transitions is possible. For problems with finite state space and finite action space, there has been a great effort in learning an $\varepsilon$-optimal policy instead of just learning an $\varepsilon$-optimal value function. In this context, a line of work [146, 147] attempted to improve the dependence on the term $1/(1-\gamma)$ in sample complexity and recently, this question is addressed in [146] by achieving an $\tilde{O}(\frac{|\mathcal{S}||\mathcal{A}|}{(1-\gamma)^3\varepsilon^2})$ upper bound that matches the lower bound from [15]. Regardless, traditional results on learning $\varepsilon$-optimal policy/value commonly scale as the product $|\mathcal{S}||\mathcal{A}|$, while our method, when reduced to this case, scales as $\max\{|\mathcal{S}|, |\mathcal{A}|\}$. In summary, Table 3.1 demonstrates that exploitation of low-rank structure consistently benefits the sample complexity of our method in the same manner for all three settings. As a caveat, we remark that our analysis does require the discounting factor $\gamma$ to be small, and we leave it as an important future direction to extend to all $\gamma$.

It is worth mentioning that designing provably efficient RL algorithms when the underlying MDP is structured has been an active research area recently. Linear structure in low-dimensional features associated with states and actions is a commonly assumed setting [178, 179, 81, 80, 3], under which the resulting sample complexity can be improved to be dependent on the (lower) dimension of the feature space instead of the original size of $\mathcal{S}$

or $\mathcal{A}$. For example, low-rank MDP studies MDP with a transition kernel $\mathcal{P}$ that is linear in the underlying feature $\phi(s,a) \in \mathbb{R}^K$. In contrast, we directly consider a low-rank structure on our learning target, the optimal $Q$-function $Q^*$, without assuming the knowledge of any feature mappings.

Lastly, we mention the recent empirical work [180] that investigates low-rank $Q^*$ with matrix estimation for finite state and action spaces. The results in [180] are solely empirical and it uses off-the-shelf ME methods. In that sense, we provide a formal framework to understand why [180] works so well, resolving the theoretical open problem raised in their work, and we provide natural generalization for continuous state and action spaces that was missing, along with a novel ME method.

**Matrix Estimation.** ME concerns recovering a low-rank $m \times n$ matrix from partial, noisy observation of it [132, 31, 32, 97, 34, 38, 49, 37]. This has been a popular topic of active research for the last few decades, which culminated in the low-rank matrix completion via convex relaxation of rank minimization [132, 31, 32]. Also, various algorithms for matrix completion/estimation – including singular value thresholding [92, 34] and nuclear-norm regularization [31, 32, 97] – have been proposed and analyzed with provable guarantees. Despite the huge success in both theory and practice, the available analysis for those existing methods only provides a handle on the error measured in Frobenius norm and a few other limited class of norms (Schatten norms, regularizing norm and its dual, etc.) under certain circumstances (Chapters 9-10, [170]). In particular, there are no satisfactory results so far that provide a control on the $\ell_\infty$ error of matrix estimation, to the best of our knowledge.

Recently, the convergence guarantees for the so-called Burer-Monteiro approach, which takes low-rank factor matrices as decision variables (also commonly referred to as "non-convex optimization" in literature), have been actively studied in pursuit of developing a computationally more efficient alternative of convex program-based approaches [64, 39]. For example, [39] provides an $\ell_\infty$ guarantee under certain setup. However, it assumes i.i.d. zero-mean noise and requires a proper initialization at the ground truth (for analysis). As a result, we were not able to use existing ME methods and their analysis in this work.

To facilitate a better understanding, Table 3.2 summarizes the setting we consider and compare it with several existing work. Clearly, in the context of applying ME to RL, a new method that provides entry-wise guarantee and handles arbitrary noise is required, and that is precisely what we provide in this work. After we elaborate our algorithmic design, in Section 3.8, we shall provide a more detailed discussions on why existing matrix estimation methods do not work and ours does, along with directions for future research.

## ■ 3.2 Markov Decision Process and Representation of $Q$-function

We consider the standard setup of infinite-horizon discounted MDP, as described by $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ in Section 1.1. We are primarily interested in continuous state space $\mathcal{S}$ and continuous action space $\mathcal{A}$, but generalization to finite spaces is readily and will be discussed. In this chapter, we consider a deterministic reward function $R(s, a)$, and to distinguish with the random case studied in Chapter 2, we use the notation $R(s, a)$ instead of $\mathcal{R}(s, a)$ throughout.

### ■ 3.2.1 MDP Regularity

We assume the existence of a generative model (i.e., a simulator) [84] and consider MDPs with the following properties:

1. (Compact domain) The state space $\mathcal{S}$ and the action space $\mathcal{A}$ are compact subsets of a Euclidean space; without loss of generality, let $\mathcal{S} = [0, 1]^{d_1}$ and $\mathcal{A} = [0, 1]^{d_2}$.

2. (Bounded reward) For every $(s, a) \in \mathcal{S} \times \mathcal{A}$, the reward $R(s, a)$ is bounded, i.e., $|R(s, a)| \leq R_{\max}$.

3. (Smoothness) The optimal $Q$-function, $Q^*$, is $\zeta$-Lipschitz with respect to the 1-product metric in $\mathcal{S}\times\mathcal{A}$, i.e., $|Q^*(s_1, a_1) - Q^*(s_2, a_2)| \leq \zeta d_{\mathcal{S}\times\mathcal{A}}\big((s_1, a_1), (s_2, a_2)\big)$ where $d_{\mathcal{S}\times\mathcal{A}}\big((s_1, a_1), (s_2, a_2)\big) = ||s_1 - s_2||_2 + \|a_1 - a_2\|_2$.

Similar to Chapter 2, we note that the bounded reward implies that for any policy $\pi$, $|V^\pi(s)| \leq V_{\max} \triangleq R_{\max}/(1 - \gamma)$ for all $s$. This yields $|Q^*(s, a)| \leq V_{\max}$, too. Again, we remark that some form of smoothness assumption is typical for learning MDPs with continuous spaces under $\ell_\infty$ guarantee. With both continuous $\mathcal{S}$ and $\mathcal{A}$, the joint Lipschitz continuity above is natural.

### ■ 3.2.2 Spectral Representation of $Q$-function

With the discussion above, $Q^* : [0, 1]^{d_1} \times [0, 1]^{d_2} \to \mathbb{R}$ is $\zeta$-Lipschitz and also bounded. This induces an integral kernel operator $K = K_{Q^*} : L^2([0, 1]^{d_1}) \to L^2([0, 1]^{d_2})$ between the spaces of square integrable functions $L^2([0, 1]^d)$ (for $d \in \{d_1, d_2\}$) endowed with the standard inner product $\langle f, g\rangle = \int_{x\in[0,1]^d} f(x)g(x)dx$, i.e., we consider the integral operator $K = K_{Q^*}$ induced by $Q^*$ that maps an integrable function $h : \mathcal{S} \to \mathbb{R}$ to $Kh : \mathcal{A} \to \mathbb{R}$ such that $Kh(a) = \int_{s\in\mathcal{S}} Q^*(s, a)h(s)ds$, $\forall a \in \mathcal{A}$. Through this lens, we obtain the following representation for $Q^*$, which follows from noticing that $K$ is a Hilbert-Schmidt operator and then applying classical results in functional analysis. The proof of Theorem 4 can be found in Appendix B.1.

**Theorem 4.** *Suppose the MDP regularity conditions (1) - (3). Then there exist a non-increasing sequence* $(\sigma_i \geq \mathbb{R}_+ : i \in \mathbb{N})$ *with* $\sum_{i=1}^{\infty} \sigma_i^2 < \infty$ *and orthonormal sets* $\{f_i \in L^2([0,1]^{d_1}) : i \in \mathbb{N}\}$ *and* $\{g_i \in L^2([0,1]^{d_2}) : i \in \mathbb{N}\}$ *such that*

$$Q^*(s,a) = \sum_{i=1}^{\infty} \sigma_i f_i(s) g_i(a), \quad \forall (s,a) \in [0,1]^{d_1} \times [0,1]^{d_2}. \tag{3.1}$$

*As a result, for any* $\delta > 0$*, there exists* $r^* = r^*(\delta) \in \mathbb{N}$ *such that for all* $r \geq r^*$*, the rank-*$r$ *approximation error satisfies* $\int_{\mathcal{S}\times\mathcal{A}} \big(\sum_{i=1}^{r} \sigma_i f_i(s) g_i(a) - Q^*(s,a)\big)^2 ds\, da = \sum_{i=r+1}^{\infty} \sigma_i^2 \leq \delta$.

**Low Rank $Q^*$.** Theorem 4 motivates us to consider low-rank $Q^*$. For any integer $r \geq 1$, we call $Q^*$ to have rank $r$ if $\sigma_i = 0$ for all $i > r$ in (3.1). More generally, we say $Q^*$ has $\delta$-approximate rank $r$ if $r^*(\delta) = r$ in Theorem 4. We focus on efficient RL for $Q^*$ with exact or approximate low rank $r$. To motivate the readers, we present an example of classical MDPs that exhibits low-rank structure in $Q^*$.

**Example 1.** The linear quadratic regulator (LQR) problem considers designing a linear controller $\pi$ for a linear dynamical system given by $s_{t+1} = As_t + Ba_t$, $a_t = \pi s_t$, via minimizing a quadratic cost (negative reward) function $R(s_t, a_t) = s_t^T E s_t + a_t^T F a_t$. Here, $s_t \in \mathbb{R}^{d_1}$ is the state of the system at time $t$, $a_t \in \mathbb{R}^{d_2}$ is the control input to the system at $t$, $A \in \mathbb{R}^{d_1 \times d_1}, B \in \mathbb{R}^{d_1 \times d_2}, \pi \in \mathbb{R}^{d_2 \times d_1}$ are matrices describing the system, and $E \in \mathbb{R}^{d_1 \times d_1}, F \in \mathbb{R}^{d_2 \times d_2}$ are symmetric positive definite matrices. According to linear-quadratic control theory [23], the value function can be expressed as $V^\pi(s_t) = s_t^T K_\pi s_t$ where $K_\pi$ is a cost matrix for policy $\pi$; thus, the $Q$-function for $\pi$ is written as

$$\begin{aligned} Q^\pi(s,a) &= R(s,a) + \gamma V^\pi\big((As + Ba)\big) \\ &= s^T(E + \gamma A^T K_\pi A)s + 2\gamma s^T A^T K_\pi B a + a^T(F + \gamma B^T K_\pi B)a. \end{aligned}$$

Letting $A^T K_\pi B = \sum_{i=1}^{r} \tau_i u_i v_i^T$ be the Singular Value Decomposition (SVD) of $A^T K_\pi B$, we can see that

$$Q^\pi(s,a) = 2\gamma \sum_{i=1}^{r} \tau_i (u_i^T s) \cdot (v_i^T a) + (s^T M_{\mathcal{S}} s) \cdot 1_{\mathcal{A}}(a) + 1_{\mathcal{S}}(s) \cdot (a^T M_{\mathcal{A}} a),$$

where $M_{\mathcal{S}} = E + \gamma A^T K_\pi A$, $M_{\mathcal{A}} = F + \gamma B^T K_\pi B$ and $1_{\mathcal{S}}$ ($1_{\mathcal{A}}$) denotes a constant-1 function on $\mathcal{S}$ ($\mathcal{A}$). It is easy to observe that $1_{\mathcal{S}}$, $s^T M_{\mathcal{S}} s$, and $\{u_i^T s\}_{i=1}^r$ form an orthogonal set in $L^2(\mathcal{S})$ as long as $\mathcal{S}$ is symmetric (i.e., $\mathcal{S} = -\mathcal{S}$). Similarly, $1_{\mathcal{A}}$, $a^T M_{\mathcal{A}} a$, and $\{v_i^T a\}_{i=1}^r$ form an orthogonal set in $L^2(\mathcal{A})$ when $\mathcal{A}$ is symmetric. Thus, the rank of $Q^\pi$ is at most $\min\{d_1, d_2\} + 2$, and so is the rank of $Q^*$, which is significantly smaller than $|\mathcal{S}| \sim 2^{d_1}$ or $|\mathcal{A}| \sim 2^{d_2}$ (after quantization).

## ■ 3.3 Reinforcement Learning via Matrix Estimation

We introduce an RL algorithm using generic ME procedures as a subroutine. We require the ME method in use to satisfy Assumption 2 (see Section 3.4) to provide meaningful performance guarantees. However, there is no known ME procedure satisfying Assumption 2 in literature. In Section 3.5, we introduce a simple ME procedure that satisfies it when $Q^*$ is exactly or approximately low-rank.

### ■ 3.3.1 A Narrative Description of the Algorithm

The RL algorithm iteratively improves estimation of $Q^*$. Each iteration consists of four steps: discretization, exploration, matrix estimation and generalization. We provide a narrative overview of the algorithm first; Algorithm 3 in Section 3.3.2 then describes the full algorithm in a pseudo-code format.

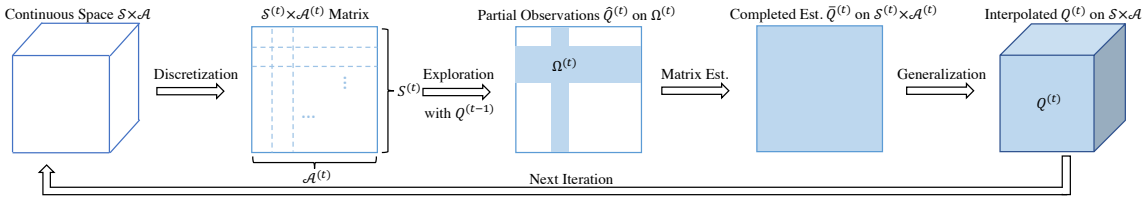



**Figure 3.1.** Iterative RL using ME: the exploration step uses estimation $Q^{(t-1)}$ from the previous iteration.

**Step 1. Discretization.** At iteration $t$, we produce $\beta^{(t)}$-nets, $\mathcal{S}^{(t)} \subset \mathcal{S}$ and $\mathcal{A}^{(t)} \subset \mathcal{A}$, for properly chosen resolution $\beta^{(t)} \in (0,1)$ that decreases with iteration $t$. In our setup, $|\mathcal{S}^{(t)}| = O\big((1/\beta^{(t)})^{d_1}\big)$, $|\mathcal{A}^{(t)}| = O\big((1/\beta^{(t)})^{d_2}\big)$. In total, this produces $|\mathcal{S}^{(t)}||\mathcal{A}^{(t)}|$ many $(s,a)$ pairs in the discretized set $\mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$.

**Step 2. Exploration.** Using estimate $Q^{(t-1)}$ over the entire $\mathcal{S} \times \mathcal{A}$ from the previous iteration, we wish to produce an improved estimate of $Q^*$ over $\mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$ through this and the next step, and then generalize it to $\mathcal{S} \times \mathcal{A}$ in Step 4. To produce an improved estimate over $\mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$ in a sample-efficient manner, we first "explore" a carefully selected subset $\Omega^{(t)} \subset \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$. Specifically, for each $(s,a) \in \Omega^{(t)}$, we obtain $N^{(t)}$ samples of independent transitions using the generative model, which results in a set of sampled next states $\{s'_i\}_{i=1,\ldots,N^{(t)}}$. We obtain an estimate $\hat{Q}^{(t)}(s,a)$ as

$$\hat{Q}^{(t)}(s,a) \leftarrow R(s,a) + \gamma \cdot \frac{1}{N^{(t)}} \sum_{i=1}^{N^{(t)}} V^{(t-1)}(s'_i), \quad \text{with } V^{(t-1)}(s) = \max_a Q^{(t-1)}(s,a). \tag{3.2}$$

**Step 3. Matrix Estimation.** Given estimates $\hat{Q}^{(t)}(s,a), \forall (s,a) \in \Omega^{(t)}$ updated in Step

2, we wish to obtain an improved estimate of $Q^*$ for the entire discretized set $\mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$. This can be viewed as a matrix estimation problem. When $Q^*$ has rank $r$ as discussed in Section 3.2, the sampled matrix $[Q^*(s,a) : s \in \mathcal{S}^{(t)}, a \in \mathcal{A}^{(t)}]$, induced by discretization, has rank at most $r$. Thus, we want to estimate the low-rank matrix by having access to noisy measurements for a subset of entries in $\Omega^{(t)} \subset \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$. Specifically, the noise in the measurements are not necessarily i.i.d. as they are coupled through $V^{(t-1)}$; thus, they are bounded but can be arbitrary. Ideally, we wish to estimate the matrix with the maximum entrywise error at a similar level as that in $\hat{Q}^{(t)}(s,a)$. This demands that the ME method in use is well-behaved in the $\ell_\infty$ sense, satisfying Assumption 2 to be stated later. While this is absent in literature, we shall describe ME methods fulfilling the desideratum in Section 3.5. As a result, we obtain improved estimates $\bar{Q}^{(t)}(s,a)$ for all $(s,a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$ after the ME step.

**Step 4. Generalization.** With estimates $\bar{Q}^{(t)}(s,a)$, $(s,a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$, we generalize to $\mathcal{S} \times \mathcal{A}$ via interpolating them. This can be achieved by any supervised learning algorithm. We simply utilize the 1-nearest neighbor: for any $(s,a) \in \mathcal{S} \times \mathcal{A}$, at the end of iteration $t$, we output $Q^{(t)}(s,a) \leftarrow \bar{Q}^{(t)}(s',a')$ where $(s',a')$ is closest to $(s,a)$ in $\mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$, with ties broken arbitrarily.

### ■ 3.3.2 Pseudo-code for the Proposed Algorithm

Below (next page) is the pseudo-code of the generic RL method described above.

## ■ 3.4 Correctness, Convergence & Sample Complexity

In this section, we state the result establishing correctness, convergence and finite sample analysis of our RL algorithm. We require a specific property, stated as Assumption 2, for the Matrix Estimation (ME) method utilized in Step 3 of the algorithm. While there is no known ME method in the literature that satisfies it, we provide a novel ME method with the desired property in Section 3.5.

### ■ 3.4.1 Matrix Estimation: a Key Premise

Recall that we describe Algorithm 3 with a generic matrix estimation subroutine used in Step 3, without specifying what ME method is used. In fact, the success of Algorithm 3 hinges on the performance of the ME method in use. For the convenience of exposition, we define $(\mathsf{C}_{\mathsf{me}}, \mathsf{c}_{\mathsf{me}})$-property of an ME method for given constants $\mathsf{C}_{\mathsf{me}}, \mathsf{c}_{\mathsf{me}} \geq 0$, which abstracts the "success" of the ME method and serves as a pivotal premise for the success of the entire RL algorithm.

**Algorithm 3** Main Algorithm: Low-rank Reinforcement Learning

**Input:** $\mathcal{S}$, $\mathcal{A}$, $\gamma$, $Q^{(0)}$, $T$, $\{\beta^{(t)}\}_{t=1,\dots,T}$, $\{N^{(t)}\}_{t=1,\dots,T}$
**Output:** $Q^{(T)}$, the $Q$-value oracle after $T$ iterations

**Initialization:** For all $s \in \mathcal{S}$, initialize the value oracle $Q^{(0)}(s)$.
**for** $t = 1, 2, \dots, T$ **do**
  `/* Step 1: Discretization of` $\mathcal{S}$ `and` $\mathcal{A}$ `*/`
  Discretize $\mathcal{S}$ and $\mathcal{A}$ so that $\mathcal{S}^{(t)}$ is a $\beta^{(t)}$-net of $\mathcal{S}$ and $\mathcal{A}^{(t)}$ is a $\beta^{(t)}$-net of $\mathcal{A}$.
  `/* Step 2: Exploration of a few` $(s,a)$ `pairs */`
  Select a subset of $(s,a)$ pairs, $\Omega^{(t)} \subseteq \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$.
  **for** $(s,a) \in \Omega^{(t)}$ **do**
    Estimate $Q^*(s,a)$ via simple lookahead based on the current value oracle $V^{(t-1)}$, i.e., query the generative model to sample $N^{(t)}$ independent transitions from $(s,a)$ and obtain an estimate $\hat{Q}^{(t)}(s,a)$ with the sampled next states $\{s'_i\}_{i=1,\dots,N^{(t)}}$:

$$\hat{Q}^{(t)}(s,a) \leftarrow R(s,a) + \gamma \cdot \frac{1}{N^{(t)}} \sum_{i=1}^{N^{(t)}} V^{(t-1)}(s'_i). \tag{3.3}$$

  **end for**
  `/* Step 3: Matrix completion to obtain` $\bar{Q}$ `from` $\hat{Q}$ `*/`
  Estimate $\bar{Q}(s,a)$ for $(s,a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$ from the data $\{\hat{Q}^{(t)}(s,a)\}_{(s,a)\in\Omega^{(t)}}$, utilizing the low-rank structure of $\bar{Q}(s,a)$, viz.,

$$\bar{Q}^{(t)} \leftarrow \text{Matrix Estimation}\big(\hat{Q}^{(t)};\ \Omega^{(t)}\big).$$

  `/* Step 4: Generalization via interpolating` $\bar{Q}$ `*/`
  Update the oracles $Q^{(t)}$ and $V^{(t)}$ by calling a subroutine that interpolates $\bar{Q}^{(t)}$ through non-parametric regression methods:

$$Q^{(t)} \leftarrow \text{Interpolation}\big(\bar{Q}^{(t)};\ \mathcal{S}^{(t)}, \mathcal{A}^{(t)}\big),$$

  and subsequently, $V^{(t)}(s) \leftarrow \max_{a\in\mathcal{A}} Q^{(t)}(s,a)$, for all $s \in \mathcal{S}$.
**end for**

**Assumption 2** ($(\mathsf{C}_{\mathsf{me}}, \mathsf{c}_{\mathsf{me}})$-property)**.** *Given finite $\mathcal{S}^{(t)} \subset \mathcal{S}$, $\mathcal{A}^{(t)} \subset \mathcal{A}$, it is possible to construct $\Omega^{(t)} \subseteq \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$ with $|\Omega^{(t)}| \le \mathsf{C}_{\mathsf{me}}\big(|\mathcal{S}^{(t)}| + |\mathcal{A}^{(t)}|\big)$ for a given constant $\mathsf{C}_{\mathsf{me}} \ge 1$ so that whenever the ME method in use takes $\{\hat{Q}^{(t)}(s,a)\}_{(s,a)\in\Omega^{(t)}}$ with $\max_{(s,a)\in\Omega^{(t)}} \big|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\big| \le \varepsilon$ as an input and outputs $\{\bar{Q}^{(t)}(s,a)\}_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}}$, the following inequality holds:*

$$\max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}} \big|\bar{Q}^{(t)}(s,a) - Q^*(s,a)\big| \le \mathsf{c}_{\mathsf{me}}\varepsilon.$$

We assume access to an ME method that satisfies $(\mathsf{C}_{\mathsf{me}}, \mathsf{c}_{\mathsf{me}})$-property. In essence, Assumption 2 ensures the $\ell_\infty$ error remains under control (to be precise, $\mathsf{c}_{\mathsf{me}}$-Lipschitz with respect to $\ell_\infty/\ell_\infty$) during the ME step, while it is stated in the language of RL for later uses. Note that Assumption 2 does not explicitly require any structure on $Q^*$, but we will

require $Q^*$ to be low-rank or approximately low-rank in order to produce an ME method satisfying the assumption, as will be discussed in Section 3.5.

### ■ 3.4.2 Correctness, Rate of Convergence & Sample Complexity of Algorithm 3

Now, we state the desired properties of the RL algorithm introduced in Section 3.3. To that end, let the algorithm start with initialization $Q^{(0)}(s,a) = 0,\ \forall (s,a) \in \mathcal{S} \times \mathcal{A}$ and hence $V^{(0)}(s) = 0,\ \forall s \in \mathcal{S}$. That is, $|Q^{(0)}(s,a) - Q^*(s,a)| \leq V_{\max},\ \forall (s,a) \in \mathcal{S} \times \mathcal{A}$. For the sake of notational brevity, we let $d_1 = d_2 = d$ in the sequel. We remark that our theorems apply equally by simply replacing $d$ with $\max\{d_1, d_2\}$.

**Theorem 5.** *Consider the RL algorithm described in Section 3.3 with ME satisfying Assumption 2. Given $\delta \in (0,1)$, there exists algorithmic choice of $\beta^{(t)}, \Omega^{(t)}, N^{(t)}$ for $1 \leq t \leq T$, so that*

$$\mathbb{P}\Big( \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \big|Q^{(t)}(s,a) - Q^*(s,a)\big| \leq (2\gamma \mathsf{c}_{\mathsf{me}})^t V_{\max}, \quad \forall 1 \leq t \leq T \Big) \geq 1 - \delta.$$

*Further, let $\gamma < \frac{1}{2\mathsf{c}_{\mathsf{me}}}$. Then, with $T = \Theta\big(\log \frac{1}{\varepsilon}\big)$ and $\widetilde{O}\big(\frac{1}{\varepsilon^{d+2}} \cdot \log \frac{1}{\delta}\big)$ number of samples, we have*

$$\mathbb{P}\Big( \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \Big|Q^{(T)}(s,a) - Q^*(s,a)\Big| \leq \varepsilon \Big) \geq 1 - \delta. \tag{3.4}$$

In the proof of Theorem 5 presented next, we choose parameters $\beta^{(t)} = \frac{V_{\max}}{8\zeta}(2\gamma\mathsf{c}_{\mathsf{me}})^t$, $|\Omega^{(t)}| = \mathsf{C}_{\mathsf{me}}(|\mathcal{S}^{(t)}| + |\mathcal{A}^{(t)}|)$ and $N^{(t)} = \frac{8}{(2\gamma\mathsf{c}_{\mathsf{me}})^{2(t-1)}} \log\big(\frac{2|\Omega^{(t)}|T}{\delta}\big)$ for $1 \leq t \leq T$. While this choice establishes the claims in Theorem 5, it is possible to achieve $\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \big|Q^{(t)}(s,a) - Q^*(s,a)\big| \leq \alpha^t V_{\max}$ for any $\alpha > \gamma\mathsf{c}_{\mathsf{me}}$ by making a more sophisticated choice. Subsequently, the conclusion for sample complexity in (3.4), can be extended for any $\gamma < \frac{1}{\mathsf{c}_{\mathsf{me}}}$. Thus, the constant $\mathsf{c}_{\mathsf{me}}$ in Assumption 2 determines the range of MDPs for which such gains can be achieved. In our analysis of the proposed ME method, $\mathsf{c}_{\mathsf{me}} \geq 1$ and indeed, we can achieve $\mathsf{c}_{\mathsf{me}} = 1$ by trivially selecting $\Omega^{(t)} = \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$, which however, does not lead to any gain in efficiency. The key challenge is to find the right balance between small $\mathsf{c}_{\mathsf{me}}$ with small $|\Omega^{(t)}|$ or $\mathsf{C}_{\mathsf{me}}$. We address this in the next sections.

### ■ 3.4.3 Proof of Theorem 5

**Helper Lemma: Error Bound for Lookahead Subroutine.** This section is devoted to the proof of Theorem 5. To this end, we first need to understand the error guarantees

for the lookahead (exploration) subroutine based on the current oracle $V^{(t-1)}$, cf. (3.2) and Line 8 of Algorithm 3. This is summarized in the following lemma.

**Lemma 10.** *Suppose that we have access to a value oracle $V : \mathcal{S} \to \mathbb{R}$ such that*

$$\sup_{s \in \mathcal{S}} \left| V(s) - V^*(s) \right| \le B.$$

*Given $(s, a) \in \mathcal{S} \times \mathcal{A}$, let $s'_1, \dots, s'_N$ be the next states of $(s, a)$ independently drawn from the generative model and let $\hat{Q}(s, a) = R(s, a) + \gamma \cdot \frac{1}{N} \sum_{i=1}^{N} V(s'_i)$. Then for any $\delta > 0$,*

$$|\hat{Q}(s, a) - Q^*(s, a)| \le \gamma \left( B + \sqrt{\frac{2V_{\max}^2}{N} \log\left(\frac{2}{\delta}\right)} \right)$$

*with probability at least $1 - \delta$.*

*Proof.* Note that $Q^*(s, a) = R(s, a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)}[V^*(s')]$ by definition of $Q^*$ and $V^*$ (cf. Bellman equation). It follows that

$$\begin{aligned} |\hat{Q}(s, a) - Q^*(s, a)| &= \gamma \left| \frac{1}{N} \sum_{i=1}^{N} V(s'_i) - \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)}[V^*(s')] \right| \\ &\le \gamma \left| \frac{1}{N} \sum_{i=1}^{N} V(s'_i) - \frac{1}{N} \sum_{i=1}^{N} V^*(s'_i) \right| + \gamma \left| \frac{1}{N} \sum_{i=1}^{N} V^*(s'_i) - \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \left[ V^*(s') \right] \right| \\ &= \frac{\gamma}{N} \sum_{i=1}^{N} \left| V(s'_i) - V^*(s'_i) \right| + \gamma \left| \frac{1}{N} \sum_{i=1}^{N} V^*(s'_i) - \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \left[ V^*(s') \right] \right|. \end{aligned} \tag{3.5}$$

By assumption, the first term in (3.5) is bounded by $\gamma B$. Meanwhile, since $|V^*(s')| \le V_{\max}$, we can apply Hoeffding's inequality to control the second term. Specifically, for any $t > 0$,

$$\mathbb{P}\left( \frac{1}{N} \sum_{i=1}^{N} V^*(s'_i) - \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \left[ V^*(s') \right] > t \right) \le \exp\left( - \frac{Nt^2}{2V_{\max}^2} \right).$$

Solving $\delta = 2 \exp\left( - \frac{Nt^2}{2V_{\max}^2} \right)$ for $t$ yields the result $t = \sqrt{\frac{2V_{\max}^2}{N} \log\left(\frac{2}{\delta}\right)}$, and this completes the proof. □

**Proof of Theorem 5: Convergence.** We prove the first statement of Theorem 5 by mathematical induction. For $t = 0$, $Q^{(0)}(s, a) \equiv 0$ and thus $|Q^{(0)}(s, a) - Q^*(s, a)| \le V_{\max}$ for all $(s, a)$. Next, we want to show that for $t = 1, \dots, T$,

$$\sup_{(s,a) \in \mathcal{S} \times \mathcal{A}} \left| Q^{(t)}(s, a) - Q^*(s, a) \right| \le \rho \sup_{(s,a) \in \mathcal{S} \times \mathcal{A}} \left| Q^{(t-1)}(s, a) - Q^*(s, a) \right|. \tag{3.6}$$

Fix $t$ and suppose that $\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}}\left|Q^{(t-1)}(s,a)-Q^*(s,a)\right|\le B^{(t-1)}$. Note that this implies $\sup_{s\in\mathcal{S}}\left|V^{(t-1)}(s)-V^*(s)\right|\le B^{(t-1)}$ because $Q^{(t-1)},Q^*$ are continuous and $\mathcal{A}$ is compact[1]. To prove the inequality in (3.6), we backtrack the updating steps in Algorithm 3.

For each $s\in\mathcal{S}$ and $a\in\mathcal{A}$, let $\hat{s}^{(t)}\in\arg\min_{s'\in\mathcal{S}^{(t)}}\|s'-s\|_2$ and $\hat{a}^{(t)}\in\arg\min_{a'\in\mathcal{A}^{(t)}}\|a'-a\|_2$. Since $\mathcal{S}^{(t)}$ is a $\beta^{(t)}$-net of $\mathcal{S}$, $\|\hat{s}^{(t)}-s\|\le\beta^{(t)}$. Likewise, $\|\hat{a}^{(t)}-a\|\le\beta^{(t)}$. As $Q^{(t)}(s,a)=\bar{Q}^{(t)}(\hat{s}^{(t)},\hat{a}^{(t)})$ and $Q^*$ is $\zeta$-Lipschitz,

$$\begin{aligned}\left|Q^{(t)}(s,a)-Q^*(s,a)\right| &= \left|\bar{Q}^{(t)}(\hat{s}^{(t)},\hat{a}^{(t)})-Q^*(s,a)\right| \\ &= \left|\bar{Q}^{(t)}(\hat{s}^{(t)},\hat{a}^{(t)})-Q^*(\hat{s}^{(t)},\hat{a}^{(t)})\right|+\left|Q^*(\hat{s}^{(t)},\hat{a}^{(t)})-Q^*(s,a)\right| \\ &\le \left|\bar{Q}^{(t)}(\hat{s}^{(t)},\hat{a}^{(t)})-Q^*(\hat{s}^{(t)},\hat{a}^{(t)})\right|+2\zeta\beta^{(t)}.\end{aligned}$$

Therefore, we obtain the following upper bound for Step 4 (interpolation):

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}}\left|Q^{(t)}(s,a)-Q^*(s,a)\right|\le\max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}}\left|\bar{Q}^{(t)}(s,a)-Q^*(s,a)\right|+2\zeta\beta^{(t)}. \tag{3.7}$$

By Assumption 2, we have the following upper bound for Step 3 (matrix estimation):

$$\max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}}\left|\bar{Q}^{(t)}(s,a)-Q^*(s,a)\right|\le\mathsf{c}_{\mathsf{me}}\max_{(s,a)\in\Omega^{(t)}}\left|\hat{Q}^{(t)}(s,a)-Q^*(s,a)\right|. \tag{3.8}$$

Lastly, applying Lemma 10 and taking union bound over $(s,a)\in\Omega^{(t)}$, we can show that

$$\max_{(s,a)\in\Omega^{(t)}}\left|\hat{Q}^{(t)}(s,a)-Q^*(s,a)\right|\le\gamma\left(B^{(t-1)}+\sqrt{\frac{2V_{\max}^2}{N^{(t)}}\log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)}\right) \tag{3.9}$$

with probability at least $1-\frac{\delta}{T}$.

Combining (3.7), (3.8) and (3.9) yields

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}}\left|Q^{(t)}(s,a)-Q^*(s,a)\right|\le B^{(t)}$$

with probability at least $1-\frac{\delta}{T}$ where

$$B^{(t)}=\gamma\mathsf{c}_{\mathsf{me}}\left(B^{(t-1)}+\sqrt{\frac{2V_{\max}^2}{N^{(t)}}\log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)}\right)+2\zeta\beta^{(t)}.$$

[1]For each $s\in\mathcal{S}$, there exist $a^{(t-1)}(s),a^*(s)\in\mathcal{A}$ such that $V^{(t-1)}(s)=Q^{(t-1)}(s,a^{(t-1)}(s))$ and $V^*(s)=Q^*(s,a^*(s))$. If $V^{(t-1)}(s)\ge V^*(s)$, then $V^{(t-1)}(s)-V^*(s)=Q^{(t-1)}(s,a^{(t-1)}(s))-Q^*(s,a^*(s))\le Q^{(t-1)}(s,a^{(t-1)}(s))-Q^*(s,a^{(t-1)}(s))$. If $V^{(t-1)}(s)<V^*(s)$, then $V^*(s)-V^{(t-1)}(s)=Q^*(s,a^*(s))-Q^{(t-1)}(s,a^{(t-1)}(s))\le Q^*(s,a^*(s))-Q^{(t-1)}(s,a^*(s))$. Therefore, $|V^{(t-1)}(s)-V^*(s)|\le\max_{a\in\{a^{(t-1)}(s),a^*(s)\}}\{Q^{(t-1)}(s,a)-Q^*(s,a)\}$.

By Assumption 2, this requires at most $|\Omega^{(t)}| = \mathsf{C}_{\mathsf{me}}\big(|\mathcal{S}^{(t)}| + |\mathcal{A}^{(t)}|\big)$. Moreover, for each $1 \le t \le T$, if we choose $\beta^{(t)} = \frac{V_{\max}}{8\zeta}(2\gamma\mathsf{C}_{\mathsf{me}})^t$ and

$$N^{(t)} = \frac{8}{(2\gamma\mathsf{C}_{\mathsf{me}})^{2(t-1)}} \log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right), \tag{3.10}$$

then $B^{(t-1)} \le (2\gamma\mathsf{C}_{\mathsf{me}})^{t-1}V_{\max}$ implies that $B^{(t)} \le (2\gamma\mathsf{C}_{\mathsf{me}})^t V_{\max}$ with probability at least $1 - \frac{\delta}{T}$.

At the beginning, we observed $|Q^{(0)}(s,a) - Q^*(s,a)| \le V_{\max}$ for all $(s,a)$, i.e., $B^{(0)} \le V_{\max}$. By taking the union bound over $t = 1, \dots, T$,

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \big|Q^{(t)}(s,a) - Q^*(s,a)\big| \le (2\gamma\mathsf{C}_{\mathsf{me}})^t V_{\max}, \quad \forall t = 1, \dots, T$$

with probability at least $1 - \delta$. □

**Proof of Theorem 5: Sample Complexity.** If $\gamma < \frac{1}{2\mathsf{C}_{\mathsf{me}}}$, then $2\gamma\mathsf{C}_{\mathsf{me}} < 1$. Let $T_\varepsilon = \left\lceil \frac{\log\left(\frac{V_{\max}}{\varepsilon}\right)}{\log\left(\frac{1}{2\gamma\mathsf{C}_{\mathsf{me}}}\right)} \right\rceil$ and observe that $(2\gamma\mathsf{C}_{\mathsf{me}})\varepsilon \le (2\gamma\mathsf{C}_{\mathsf{me}})^{T_\varepsilon}V_{\max} \le \varepsilon$. For each $t, 1 \le t \le T$, we query $\hat{Q}^{(t)}(s,a)$ for $(s,a) \in \Omega^{(t)}$, each of which requires exploring $N^{(t)}$ samples. Therefore, the total sample complexity of Algorithm 3 with $T = T_\varepsilon$ is $\sum_{t=1}^{T_\varepsilon} |\Omega^{(t)}|N^{(t)}$.

By standard argument on covering number, we can see that $|\mathcal{S}^{(t)}|, |\mathcal{A}^{(t)}| \le C'\big(\frac{1}{\beta^{(t)}}\big)^d = C'\big(\frac{8\zeta}{V_{\max}}\big)^d\big(2\gamma\mathsf{C}_{\mathsf{me}}\big)^{-dt}$ for some absolute constant $C' > 0$. This is an increasing function of $t$ and hence, $|\Omega^{(t)}| = \mathsf{C}_{\mathsf{me}}\big(|\mathcal{S}^{(t)}| + |\mathcal{A}^{(t)}|\big)$ and $N^{(t)}$ as described in (3.10) are also increasing with respect to $t$.

Observe that $\beta^{(T_\varepsilon)} = \frac{V_{\max}}{8\zeta}(2\gamma\mathsf{C}_{\mathsf{me}})^{T_\varepsilon} \ge \frac{2\gamma\mathsf{C}_{\mathsf{me}}}{8\zeta}\varepsilon$. Hence, $|\mathcal{S}^{(T_\varepsilon)}|, |\mathcal{A}^{(T_\varepsilon)}| \le C'\big(\frac{8\zeta}{2\gamma\mathsf{C}_{\mathsf{me}}}\big)^d \frac{1}{\varepsilon^d}$. Therefore, the overall number of samples utilized by the algorithm is

$$\begin{aligned}
\sum_{t=1}^{T_\varepsilon} |\Omega^{(t)}|N^{(t)} &\le T_\varepsilon |\Omega^{(T_\varepsilon)}|N^{(T_\varepsilon)} \\
&\le T_\varepsilon \cdot \mathsf{C}_{\mathsf{me}}\big(|\mathcal{S}^{(T_\varepsilon)}| + |\mathcal{A}^{(T_\varepsilon)}|\big) \cdot \frac{8}{(2\gamma\mathsf{C}_{\mathsf{me}})^{2(T_\varepsilon-1)}} \log\left(\frac{2\mathsf{C}_{\mathsf{me}}\big(|\mathcal{S}^{(T_\varepsilon)}| + |\mathcal{A}^{(T_\varepsilon)}|\big)T_\varepsilon}{\delta}\right) \\
&\le T_\varepsilon \cdot 2\mathsf{C}_{\mathsf{me}}C'\left(\frac{8\zeta}{2\gamma\mathsf{C}_{\mathsf{me}}}\right)^d \frac{1}{\varepsilon^d} \cdot 8\left(\frac{V_{\max}}{\varepsilon}\right)^2 \log\left(\frac{4\mathsf{C}_{\mathsf{me}}C'T_\varepsilon}{\delta}\left(\frac{8\zeta}{2\gamma\mathsf{C}_{\mathsf{me}}}\right)^d \frac{1}{\varepsilon^d}\right) \\
&= 16\mathsf{C}_{\mathsf{me}}C'V_{\max}^2\left(\frac{8\zeta}{2\gamma\mathsf{C}_{\mathsf{me}}}\right)^d \cdot \frac{T_\varepsilon}{\varepsilon^{d+2}} \cdot \log\left(4\mathsf{C}_{\mathsf{me}}C'\left(\frac{8\zeta}{2\gamma\mathsf{C}_{\mathsf{me}}}\right)^d \cdot \frac{T_\varepsilon}{\varepsilon^d} \cdot \frac{1}{\delta}\right). \qquad (3.11)
\end{aligned}$$

Since $T_\varepsilon = \left\lceil \frac{\log\left(\frac{V_{\max}}{\varepsilon}\right)}{\log\left(\frac{1}{2\gamma\mathsf{C}_{\mathsf{me}}}\right)} \right\rceil = O\big(\log\frac{1}{\varepsilon}\big)$, it follows from (3.11) that the overall sample complexity scales as $O\left(\frac{1}{\varepsilon^{d+2}}\log\frac{1}{\varepsilon} \cdot \left(\log\frac{1}{\varepsilon} + \log\frac{1}{\delta}\right)\right)$. This completes the proof of Theorem 5. □

## ■ 3.5 Matrix Estimation Satisfying Assumption 2

We introduce a matrix estimation method satisfying Assumption 2 which is required for the success of our RL algorithm as in Theorem 5. For the ease of illustration, we start with describing it for the rank-1 setting (Section 3.5.1), then generalize it for $Q^*$ with generic rank $r \geq 1$ (Section 3.5.2) and finally for the approximate rank-$r$ setting with full generality (Section 3.5.3). Technical proofs for all the settings are deferred to Section 3.6 for a streamlined presentation.

### ■ 3.5.1 Matrix Estimation for $Q^*$ with Rank 1: a Warm-up

Consider $Q^*$ with rank 1. That is, there exist $f : \mathcal{S} \to \mathbb{R}$ and $g : \mathcal{A} \to \mathbb{R}$ so that $Q^*(s,a) = f(s)g(a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. For the ease of exposition, we assume $R(s,a) \in [R_{\min}, R_{\max}]$ with $R_{\min} > 0$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ in this warm-up only. Subsequently, $Q^*(s,a) \geq V_{\min} \triangleq \frac{R_{\min}}{1-\gamma}$, $\forall (s,a)$.

**Matrix Estimation Algorithm.** For $t \geq 1$, consider a discretization of state, action spaces, $\mathcal{S}^{(t)} \subset \mathcal{S}$, $\mathcal{A}^{(t)} \subset \mathcal{A}$. Let $Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)})$ be the $|\mathcal{S}^{(t)}| \times |\mathcal{A}^{(t)}|$ matrix induced by restricting $Q^*$ to $\mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$. Since $Q^*$ is rank 1, it follows that $Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)}) = FG^T$ where $F = [f(s) : s \in \mathcal{S}^{(t)}] \in \mathbb{R}^{|\mathcal{S}^{(t)}| \times 1}$ and $G = [g(a) : a \in \mathcal{A}^{(t)}] \in \mathbb{R}^{|\mathcal{A}^{(t)}| \times 1}$. Therefore, we can estimate $Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)})$ by estimating $F, G$.

Now we describe the selection of $\Omega^{(t)}$ such that $|\Omega^{(t)}| = |\mathcal{S}^{(t)}| + |\mathcal{A}^{(t)}| - 1$. To that end, we first choose an *anchor* element $s^\sharp \in \mathcal{S}^{(t)}$ and $a^\sharp \in \mathcal{A}^{(t)}$. Then, let $\Omega^{(t)} = \{(s,a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)} : s = s^\sharp \text{ or } a = a^\sharp\}$. With access to $\{\hat{Q}^{(t)}(s,a) : (s,a) \in \Omega^{(t)}\}$, our ME method produces estimates for all $(s,a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$ as

$$\bar{Q}^{(t)}(s,a) = \frac{\hat{Q}^{(t)}(s,a^\sharp)\hat{Q}^{(t)}(s^\sharp,a)}{\hat{Q}^{(t)}(s^\sharp,a^\sharp)}.$$

**Satisfaction of Assumption 2.** For the algorithm described above, we state the following proposition which verifies that Assumption 2 is satisfied with $\mathsf{C}_{\mathsf{me}} = 1$ and $\mathsf{c}_{\mathsf{me}} = 7\frac{R_{\max}}{R_{\min}}$.

**Proposition 2.** *For $\varepsilon \leq \frac{1}{2}V_{\min}$, suppose that $\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| \leq \varepsilon$. Then the estimate produced by the above ME algorithm satisfies*

$$\max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}} \left|\bar{Q}^{(t)}(s,a) - Q^*(s,a)\right| \leq 7\frac{R_{\max}}{R_{\min}}\varepsilon.$$

Proposition 2 implies that when $Q^*$ has rank 1, our simple ME method described above satisfies $\left(1, 7\frac{R_{\max}}{R_{\min}}\right)$-property for $\varepsilon \leq \frac{1}{2}V_{\min}$. We remark that for any $c \in (0,1)$, one can show that the method fulfills $\left(1, \mathsf{c}_{\mathsf{me}}\right)$-property with $\mathsf{c}_{\mathsf{me}} = \frac{3+c}{1-c}\frac{R_{\max}}{R_{\min}}$ for all $\varepsilon \leq cV_{\min}$. By

replacing Assumption 2 in Theorem 5 with Proposition 2, we obtain convergence and sample complexity guarantees for the rank-1 setup as stated in Theorem 6 below. We refer readers to Section 3.6.1 for proofs of Proposition 2 and Theorem 6.

**Theorem 6.** *Let $Q^*$ be rank 1. Consider the RL algorithm (cf. Section 3.3) with the Matrix Estimation method as described in Section 3.5.1. If $\gamma < \frac{R_{\min}}{14R_{\max}}$, then the following statements hold.*

1. *For any $\delta > 0$, we have*

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(t)}(s,a) - Q^*(s,a)\right| \leq \left(\frac{14R_{\max}}{R_{\min}}\gamma\right)^t V_{\max}, \quad \forall\ 1 \leq t \leq T,$$

   *with probability at least $1-\delta$ by choosing algorithmic parameters $\beta^{(t)}, N^{(t)}$ appropriately.*

2. *Further, given $\varepsilon > 0$, it suffices to set $T = \Theta(\log \frac{1}{\varepsilon})$ and use $\tilde{O}(\frac{1}{\varepsilon^{d+2}} \cdot \log \frac{1}{\delta})$ number of samples to achieve*

$$\mathbb{P}\left(\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(T)}(s,a) - Q^*(s,a)\right| \leq \varepsilon\right) \geq 1 - \delta.$$

### ■ 3.5.2 Matrix Estimation for $Q^*$ with Rank $r$

Based on the intuition developed in Section 3.5.1, we consider a more general rank-$r$ setup. For notational convenience, given $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $\mathcal{S}' \subset \mathcal{S}, \mathcal{A}' \subset \mathcal{A}$, we let $Q(\mathcal{S}', \mathcal{A}')$ denote the $|\mathcal{S}'| \times |\mathcal{A}'|$ matrix $\left[Q(s,a) : \ (s,a) \in \mathcal{S}' \times \mathcal{A}'\right]$, whose entries are indexed by $(s,a) \in \mathcal{S}' \times \mathcal{A}'$.

The central idea is the same as before: although $Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)}) \in \mathbb{R}^{m\times n}$ is an array of $mn$ real numbers, it has only $r(m+n-r)$ degrees of freedom with $r$-dimensional row and column spaces, when $\text{rank}(Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)})) = r \leq \min\{m,n\}$; as a result, one can successfully restore $Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)})$ by exploring only $r$ entire rows and columns. There is, however, a small caveat that the $r$ rows and $r$ columns should be carefully chosen so that they are not degenerate, i.e., the $r$ rows span the entire row space of $Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)})$ (the $r$ columns span the entire column space of $Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)})$, respectively). Towards this end, we first define the notion of anchor states and actions.

**Definition 1.** *(Anchor states and actions) A set of states $\mathcal{S}^\sharp = \{s_i^\sharp\}_{i=1}^{R_s} \subset \mathcal{S}$ and actions $\mathcal{A}^\sharp = \{a_i^\sharp\}_{i=1}^{R_a} \subset \mathcal{A}$ for some $R_s, R_a$ are called anchor states and anchor actions for $Q^*$ if rank $Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp) = r$.*

That is, there are $r$ states in the set $\mathcal{S}^\sharp$ such that $Q^*(s, \mathcal{A}^\sharp), s \in \mathcal{S}^\sharp$ are linearly independent. In other words, $\mathcal{S}^\sharp$ contains states with sufficiently diverse performance on actions $\mathcal{A}^\sharp$.

Likewise, a similar interpretation holds for $\mathcal{A}^\sharp$ if we look at the columns of $Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)$.

Indeed, $\mathcal{S}^\sharp$ and $\mathcal{A}^\sharp$ will be applied to construct our exploration sets and we want them to have small size. Finding only a few diverse states and actions is arguably easy in practice — in fact, for several stochastic control tasks experimented in Section 3.7, we simply pick a few states and actions that are far from each other in their respective metric spaces. We remark that assuming some "anchor" elements (i.e., elements having some special, relevant properties) is common in feature-based reinforcement learning [178, 53] or matrix factorization such as topic modeling [10].

**Matrix Estimation Algorithm.** We select anchor states $\mathcal{S}^\sharp \subset \mathcal{S}$, anchor actions $\mathcal{A}^\sharp \subset \mathcal{A}$ and fix them throughout all iterations $1 \le t \le T$. As before, we select appropriate $\beta^{(t)}$-nets $\mathcal{S}^{(t)}$ and $\mathcal{A}^{(t)}$ and augment them with the anchor states and actions: $\bar{\mathcal{S}}^{(t)} \leftarrow \mathcal{S}^{(t)} \cup \mathcal{S}^\sharp$ and $\bar{\mathcal{A}}^{(t)} \leftarrow \mathcal{A}^{(t)} \cup \mathcal{A}^\sharp$. For iteration $1 \le t \le T$, we let

$$\Omega^{(t)} = \{(s,a) \in \bar{\mathcal{S}}^{(t)} \times \bar{\mathcal{A}}^{(t)} : s \in \mathcal{S}^\sharp \text{ or } a \in \mathcal{A}^\sharp\}$$

be the exploration set.

Given $\hat{Q}^{(t)}(s,a)$ for $(s,a) \in \Omega^{(t)}$, our ME method produces estimates for all $(s,a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$ as

$$\bar{Q}^{(t)}(s,a) = \hat{Q}^{(t)}(s, \mathcal{A}^\sharp)\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger \hat{Q}^{(t)}(\mathcal{S}^\sharp, a), \tag{3.12}$$

where $\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger$ denotes the Moore-Penrose pseudoinverse of $\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)$. With the choice of $R_s = R_a = r$ (or a constant multiple of $r$, e.g. $O(r)$), the size of $\Omega^{(t)}$ is at most $r\big(|\bar{\mathcal{S}}^{(t)}| + |\bar{\mathcal{A}}^{(t)}| - r\big) \ll |\bar{\mathcal{S}}^{(t)}||\bar{\mathcal{A}}^{(t)}|$.

**Satisfaction of Assumption 2.** For given matrix $X \in \mathbb{R}^{m\times n}$, we denote by $\sigma_i(X)$ its $i$-th largest singular value, i.e., $\sigma_1(X) \ge \sigma_2(X) \ge \cdots \ge \sigma_{\min(m,n)}(X) \ge 0$. We state the following guarantee, which verifies that the matrix estimation algorithm described above satisfies Assumption 2.

**Proposition 3.** *Let $\Omega^{(t)}$ and $\bar{Q}^{(t)}$ as described above. For any $\varepsilon \le \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)$, if $\max_{(s,a)\in\Omega^{(t)}} \big|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\big| \le \varepsilon$, then*

$$\max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}} \big|\bar{Q}^{(t)}(s,a) - Q^*(s,a)\big| \le c(r; \mathcal{S}^\sharp, \mathcal{A}^\sharp)\varepsilon,$$

*where*

$$c(r; \mathcal{S}^\sharp, \mathcal{A}^\sharp) = \left(6\sqrt{2}\left(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\right) + 2(1+\sqrt{5})\left(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\right)^2\right) V_{\max}$$

Proposition 3 implies when $Q^*$ has rank $r$, our ME method satisfies $\big(\max\{|\mathcal{S}^\sharp|, |\mathcal{A}^\sharp|\}, c(r; \mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)$-

property for $\varepsilon \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)$. Hence, we obtain Theorem 7 below as a corollary of Theorem 5. That is, we obtain the desired convergence result and sample complexity $\tilde{O}(\frac{1}{\varepsilon^{d+2}} \cdot \log \frac{1}{\delta})$ to achieve $\varepsilon$ error with the output $Q^{(T)}$. We remark that here, the hidden constant in $\tilde{O}(\cdot)$ depends on $|\mathcal{S}^\sharp|$ and $|\mathcal{A}^\sharp|$. Based on the definition, the minimal values of $|\mathcal{S}^\sharp|$ and $|\mathcal{A}^\sharp|$ are both $r$. As common in the ME literature, we generally considers low-rank problems with a small value $r$ and treats related quantities as constants. The proofs of Proposition 3 and Theorem 7 are provided in Section 3.6.2.

**Theorem 7.** *Let $Q^*$ have rank $r$. Consider the RL algorithm (cf. Section 3.3) with the Matrix Estimation method as described in Section 3.5.2. If $\gamma \leq \frac{1}{2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}$, then the following statements hold.*

1. *For any $\delta > 0$, we have*

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(t)}(s,a) - Q^*(s,a)\right| \leq \big(2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\big)^t V_{\max}, \quad \text{for all } t = 1,\ldots,T$$

   *with probability at least $1-\delta$ by choosing algorithmic parameters $\beta^{(t)}, N^{(t)}$ appropriately.*

2. *Further, given $\varepsilon > 0$, it suffices to set $T = \Theta(\log \frac{1}{\varepsilon})$ and use $\tilde{O}(\frac{1}{\varepsilon^{d+2}} \cdot \log \frac{1}{\delta})$ number of samples to achieve*

$$\mathbb{P}\left(\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(T)}(s,a) - Q^*(s,a)\right| \leq \varepsilon\right) \geq 1-\delta.$$

**Remark.** *(Discrete spaces)* It is clear that our algorithm and analysis also apply to low-rank $Q^*$ defined over discrete spaces (as also mentioned in Table 3.1 at the beginning of this chapter). We summarize results for (1) continuous $\mathcal{S}$ and finite $\mathcal{A}$; (2) finite $\mathcal{S}$ and finite $\mathcal{A}$ as direct corollaries of Theorem 7 in Appendix B.2.

### ■ 3.5.3 Matrix Estimation for $Q^*$ with Approximate Rank $r$

In Section 3.5.2, we considered the setup where the underlying $Q^*$ has an exact rank $r$. However, it may not be feasible to hope for exact low-rank structure in practice. Hence, it is desirable to seek methods that are reasonably robust to approximation error when $Q^*$ can be well approximated by the first few spectral components. We show that our ME method has such an appealing property.

Given $r > 0$ as a parameter, let $Q^*_r$ denote the best rank-$r$ approximation of $Q^*$ in the $L^2$-sense so that $Q^*_r(s,a) = \sum_{i=1}^r \sigma_i f_i(s) g_i(a)$ (cf. Theorem 4). Denote by $\xi_r \triangleq \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} |Q^*_r(s,a) - Q^*(s,a)|$ the model bias due to approximation. We introduce the

notion of $r$-anchor states/actions that generalizes the notion of anchor states and actions in Definition 1.

**Definition 2.** *($r$-Anchor states and actions) A set of states $\mathcal{S}^\sharp = \{s_i^\sharp\}_{i=1}^{R_s} \subset \mathcal{S}$ and actions $\mathcal{A}^\sharp = \{a_i^\sharp\}_{i=1}^{R_a} \subset \mathcal{A}$ for some $R_s, R_a$ are called $r$-anchor states and $r$-anchor actions for $Q^*$ if rank $Q_r^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp) = r$ for a positive integer $r$.*

It is easy to see that if $\mathcal{S}^\sharp$ and $\mathcal{A}^\sharp$ are $r$-anchor states/actions for $Q^*$, then they are $r'$-anchor states/actions for $Q^*$ for all $r' \leq r$.

**Matrix Estimation Algorithm.** The algorithm remains the same as the exact rank-$r$ case in Section 3.5.2, except that we select $\mathcal{S}^\sharp \subset \mathcal{S}$ and $\mathcal{A}^\sharp \subset \mathcal{A}$ to be $r$-anchor states and actions.

**Theoretical Guarantee for Approximate Rank-$r$ Setup.** Previously, we imposed some regularity assumptions on $Q^*$, but the truncated function $Q_r^*$ is not guaranteed to inherit the regularity properties. Here, we additionally assume that (i) $\|Q_r^*\|_\infty \leq V_{\max}$ and (ii) $Q_r^*$ is $\zeta$-Lipschitz, for the convenience of exposition.

At a high level, our analysis is simple: for a given parameter $r > 0$, we treat $Q_r^*$ as the true function and repeat our analysis for the rank-$r$ setup. Of course, there will be an additional bias, $Q_r^*(s,a) - Q^*(s,a)$, incurred by this substitution which requires careful tracking at each iteration. We formalize this argument in Proposition 4 and Theorem 8.

**Proposition 4.** *Let $\Omega^{(t)}$ and $\bar{Q}^{(t)}$ be as described above. Given a positive integer $r > 0$, let $\mathcal{S}^\sharp$ and $\mathcal{A}^\sharp$ be some $r$-anchor states and actions for $Q^*$.*

*For any $\varepsilon \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q_r^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)$, if $\max_{(s,a)\in\Omega^{(t)}} \big|\hat{Q}^{(t)}(s,a) - Q_r^*(s,a)\big| \leq \varepsilon$, then*

$$\max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}} \big|\bar{Q}^{(t)}(s,a) - Q_r^*(s,a)\big| \leq \phi_c(r; \mathcal{S}^\sharp, \mathcal{A}^\sharp)\varepsilon,$$

*where*

$$\phi_c(r; \mathcal{S}^\sharp, \mathcal{A}^\sharp) := \left(6\sqrt{2}\left(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r\big(Q_r^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\right) + 2(1+\sqrt{5})\left(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r\big(Q_r^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\right)^2\right)V_{\max}. \quad (3.13)$$

With Proposition 4 at hand, we can obtain the following theorem as a corollary of Theorem 5 for the approximate rank-$r$ setup. The theorem guarantees that when the model bias $\|Q_r^* - Q^*\|_\infty$ is sufficiently small, we obtain convergence and sample complexity results similar to the rank-$r$ setting with an additive error induced by the model bias.

**Theorem 8.** *Consider the approximate rank-$r$ setting in this section and the RL algorithm (cf. Section 3.3) with the Matrix Estimation method as described above. Given a positive*

*integer* $r$*, if* $\gamma \leq \frac{1}{2\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}$ *and* $\xi_r \leq \min\left\{\frac{\sigma_r\left(Q_r^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\right)}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}+(1+\frac{1}{V_{\max}})\sigma_r\left(Q_r^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\right)}, \frac{3}{2}V_{\max}\right\}$*, then the following statements hold.*

1. *For any* $\delta > 0$*, with probability at least* $1-\delta$*, the following inequality holds for all* $t = 1, \ldots, T$

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(t)}(s,a) - Q^*(s,a)\right| \leq \left(2\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\right)^t V_{\max} + (1+\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma)\xi_r \sum_{i=1}^{t}\left(\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\right)^{i-1}$$

   *by choosing algorithmic parameters* $\beta^{(t)}, N^{(t)}$ *appropriately.*

2. *Further, given* $\varepsilon > 0$*, it suffices to set* $T = \Theta(\log\frac{1}{\varepsilon})$ *and use* $\tilde{O}(\frac{1}{\varepsilon^{d+2}}\cdot\log\frac{1}{\delta})$ *number of samples to achieve*

$$\mathbb{P}\left(\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(T)}(s,a) - Q^*(s,a)\right| \leq \varepsilon + \frac{1+\gamma\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}{1-\gamma\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}\xi_r\right) \geq 1-\delta.$$

The proof of Theorem 8 is provided in Section 3.6.3. Theorem 8 establishes the robustness of our method. When the approximation error $\xi_r$ is not too large, with high probability, we obtain estimate of $Q^*$ that is within $\ell_\infty$ error $\varepsilon + \frac{1+\gamma\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}{1-\gamma\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}\xi_r$. Again, the algorithm only efficiently utilizes $\tilde{O}(\frac{1}{\varepsilon^{d+2}}\cdot\log\frac{1}{\delta})$ number of samples. Overall, the results on the approximate rank-$r$ setting justifies the soundness of our approach, from both theoretical and practical perspectives.

**Reduction to Exact Rank-$r$ Case.** Note that the exact rank-$r$ setting is a special case of the approximate rank-$r$ setup with $\xi_s = 0$ for all $s \geq r$. Therefore, Theorem 8 applies to rank-$r$ setup discussed in Section 3.5.2. $\phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)$ in (3.13) reduces to $c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)$ defined in Proposition 3. As a result, we can apply Theorem 8 to arrive at the same conclusion with Theorem 7.

## ■ 3.6 Technical Results of the Proposed ME Method

In this section, we provide the details related to the technical results presented in Section 3.5.

### ■ 3.6.1 Rank$(Q^*) = 1$

We prove Proposition 2 and Theorem 6 for the case of a rank-1 $Q^*$ here.

**Proof of Proposition 2.** First, we note that for any $(s,a) \in \mathcal{S} \times \mathcal{A}$,

$$Q^*(s,a) = f(s)g(a) = \frac{f(s)g(a^\sharp)f(s^\sharp)g(a)}{f(s^\sharp)g(a^\sharp)} = \frac{Q^*(s,a^\sharp)Q^*(s^\sharp,a)}{Q^*(s^\sharp,a^\sharp)}.$$

We assumed that $\left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| \le \varepsilon$ for all $(s,a) \in \Omega^{(t)}$. Since $(s,a^\sharp)$, $(s^\sharp,a)$, $(s^\sharp,a^\sharp) \in \Omega^{(t)}$,

$$\bar{Q}^{(t)}(s,a) \le \frac{\left(1+\frac{\varepsilon}{Q^*(s,a^\sharp)}\right)\left(1+\frac{\varepsilon}{Q^*(s^\sharp,a)}\right)}{1-\frac{\varepsilon}{Q^*(s^\sharp,a^\sharp)}} Q^*(s,a) \le \left(1+\frac{\varepsilon}{V_{\min}}\right)^2\left(1+\frac{2\varepsilon}{V_{\min}}\right)Q^*(s,a).$$

The last inequality follows from that $\frac{1}{1-x} \le 1+2x$ for $0 \le x \le \frac{1}{2}$ and that $\varepsilon \le \frac{1}{2}V_{\min} \le \min\{Q^*(s,a^\sharp), Q^*(s^\sharp,a), Q^*(s^\sharp,a^\sharp)\}$. Therefore,

$$\begin{aligned}\bar{Q}^{(t)}(s,a) - Q^*(s,a) &\le \left[4\left(\frac{\varepsilon}{V_{\min}}\right) + 5\left(\frac{\varepsilon}{V_{\min}}\right)^2 + 2\left(\frac{\varepsilon}{V_{\min}}\right)^3\right]Q^*(s,a) \\ &\le 7Q^*(s,a)\frac{\varepsilon}{V_{\min}} \le 7\frac{V_{\max}}{V_{\min}}\varepsilon.\end{aligned}$$

In a similar manner,

$$\bar{Q}^{(t)}(s,a) \ge \frac{\left(1-\frac{\varepsilon}{Q^*(s,a^\sharp)}\right)\left(1-\frac{\varepsilon}{Q^*(s^\sharp,a)}\right)}{1+\frac{\varepsilon}{Q^*(s^\sharp,a^\sharp)}} Q^*(s,a) \ge \left(1-\frac{\varepsilon}{V_{\min}}\right)^2\left(1-\frac{\varepsilon}{Q^*(s^\sharp,a^\sharp)}\right)Q^*(s,a)$$

because $\frac{1}{1+x} \ge 1-x$ for $0 \le x \le \frac{1}{2}$, and thus,

$$\bar{Q}^{(t)}(s,a) - Q^*(s,a) \ge \left[-3\left(\frac{\varepsilon}{V_{\min}}\right) + 3\left(\frac{\varepsilon}{V_{\min}}\right)^2 - \left(\frac{\varepsilon}{V_{\min}}\right)^3\right]Q^*(s,a) \ge -\frac{7}{4}\frac{V_{\max}}{V_{\min}}\varepsilon.$$

Therefore, for all $(s,a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$, $\left|\bar{Q}^{(t)}(s,a) - Q^*(s,a)\right| \le 7\frac{V_{\max}}{V_{\min}}\varepsilon = 7\frac{R_{\max}}{R_{\min}}\varepsilon$. This completes the proof of Proposition 2. □

**Proof of Theorem 6.** The proof is basically the same as the proof of Theorem 5, with the assumption on the matrix estimation oracle (i.e., Assumption 2) replaced with the explicit guarantee provided in Proposition 2. The only subtlety comes from that Proposition 2 is a "local" guarantee that holds only for $\varepsilon \le \frac{1}{2}V_{\min}$ whereas Assumption 2 is a global condition that holds for any $\varepsilon$. This requires us to ensure $\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| \le \frac{1}{2}V_{\min}$ for all $t = 1, \dots, T$, but the argument in the proof of Theorem 5 itself remains valid.

To that end, we make exactly the same choice of algorithmic parameters $\beta^{(t)}, N^{(t)}$ as

$$\beta^{(t)} = \frac{V_{\max}}{8\zeta}(2\gamma\mathsf{c}_{\mathsf{me}})^t \quad \text{and} \quad N^{(t)} = \frac{8}{(2\gamma\mathsf{c}_{\mathsf{me}})^{2(t-1)}}\log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right) \tag{3.14}$$

with $\mathsf{c}_{\mathsf{me}} = \frac{7R_{\max}}{R_{\min}}$ as suggested in Proposition 2 and $|\Omega^{(t)}| = \mathsf{C}_{\mathsf{me}}(|\mathcal{S}^{(t)}|+|\mathcal{A}^{(t)}|)$ with $\mathsf{C}_{\mathsf{me}} = 1$. To complete the proof, it suffices to show that $\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| \le \frac{1}{2}V_{\min}$ for all $t$.

We establish this via mathematical induction. For $0 \le t \le T$, let $B^{(t)} := \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(t)}(s,a) - Q^*(s,a)\right|$. We can see that if $B^{(t-1)} \le \left(\frac{14R_{\max}}{R_{\min}}\gamma\right)^{t-1} V_{\max}$, then with probability at least $1-\frac{\delta}{T}$, the following two inequalities hold:

1. $\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| \le \frac{1}{2}V_{\min}$, and
2. $B^{(t)} \le \frac{14R_{\max}}{R_{\min}}\gamma B^{(t-1)}$.

The first inequality follows from Lemma 10 (see also (3.9)): with probability at least $1-\frac{\delta}{T}$,

$$\begin{aligned}\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| &\le \gamma\left(B^{(t-1)} + \sqrt{\frac{2V_{\max}^2}{N^{(t)}}\log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)}\right) \le \frac{3}{2}\gamma B^{(t-1)} \\ &\le \frac{3}{2}\gamma V_{\max} \le \frac{3}{2}\frac{R_{\min}}{14R_{\max}}V_{\max} = \frac{3}{28}V_{\min} \\ &\le \frac{1}{2}V_{\min}.\end{aligned}$$

Also, the second inequality follows from the same argument as in the proof of Theorem 5, cf. (3.7), (3.8) and (3.9).

It remains to certify that $B^{(t)} \le \left(\frac{14R_{\max}}{R_{\min}}\gamma\right)^t V_{\max}$ for $t = 0,\dots,T-1$. First of all, $Q^{(0)}(s,a) \equiv 0$ by assumption, and hence, $B^{(0)} \le V_{\max}$. Thus, by the second inequality and the condition on $\gamma$, $B^{(t)} \le \left(\frac{14R_{\max}}{R_{\min}}\gamma\right)B^{(t-1)} \le \cdots \le \left(\frac{14R_{\max}}{R_{\min}}\gamma\right)^t B^{(0)} \le \left(\frac{14R_{\max}}{R_{\min}}\gamma\right)^t V_{\max}$ for all $t = 0,\dots,T-1$ and the proof is complete. □

## ■ 3.6.2 Rank$(Q^*) = r$

In this section, we prove results related to the exact rank-$r$ case. We begin with the proof of Proposition 3 and then the proof of Theorem 7.

**Proof of Proposition 3.** We start with a helper lemma below.

**Lemma 11.** *Let* $M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$. *If rank* $A$ = *rank* $M$, *then* $D = CA^\dagger B$.

*Proof.* Since $\operatorname{rank}_{\text{row}} \begin{bmatrix} A & B \end{bmatrix} \ge \operatorname{rank}_{\text{row}} A = \operatorname{rank} A = \operatorname{rank} M = \operatorname{rank}_{\text{row}} M$, there exists a matrix $P$ such that $\begin{bmatrix} C & D \end{bmatrix} = P\begin{bmatrix} A & B \end{bmatrix}$. Also, observe that $\operatorname{rank}_{\text{col}} \begin{bmatrix} A & B \end{bmatrix} \le \operatorname{rank}_{\text{col}} M = \operatorname{rank} M = \operatorname{rank} A = \operatorname{rank}_{\text{col}} A$. That is, the column space of $B$ is a subspace of the column space of $A$. It follows that $AA^\dagger A = A$ and $AA^\dagger B = B$ because the left multiplication of

$AA^\dagger$ is the projection on the column space of $A$. We obtain

$$\begin{bmatrix} C & D \end{bmatrix} = P \begin{bmatrix} A & B \end{bmatrix} = P \begin{bmatrix} AA^\dagger A & AA^\dagger B \end{bmatrix} = \begin{bmatrix} PA & PAA^\dagger B \end{bmatrix}.$$

Therefore, $PA = C$ and $D = PAA^\dagger B = CA^\dagger B$. □

Now, we are ready to prove Proposition 3. First, we observe that for any $(s, a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$

$$Q^*(s,a) = Q^*(s, \mathcal{A}^\sharp)\big[Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger Q^*(\mathcal{S}^\sharp, a). \tag{3.15}$$

This can be verified by applying Lemma 11 to $M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \in \mathbb{R}^{|\bar{\mathcal{S}}^{(t)}| \times |\bar{\mathcal{A}}^{(t)}|}$ where

$$\begin{aligned} A &= Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp), & B &= Q^*(\mathcal{S}^\sharp, \mathcal{A}^{(t)}), \\ C &= Q^*(\mathcal{S}^{(t)}, \mathcal{A}^\sharp), & D &= Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)}). \end{aligned}$$

Here, rank $A = r = $ rank $M$ by definition of anchor states/actions and the fact that $Q^*$ has rank $r$. Hence,

$$Q^*(\mathcal{S}^{(t)}, \mathcal{A}^{(t)}) = D = CA^\dagger B = Q^*(\mathcal{S}^{(t)}, \mathcal{A}^\sharp)\big[Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger Q^*(\mathcal{S}^\sharp, \mathcal{A}^{(t)}).$$

Next, we fix $(s, a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$ and consider the error $\bar{Q}^{(t)}(s,a) - Q^*(s,a)$. According to the definition of $\bar{Q}^{(t)}(s,a)$ (cf. (3.12)) and (3.15),

$$\begin{aligned} \bar{Q}^{(t)}(s,a) - Q^*(s,a) &= \hat{Q}^{(t)}(s, \mathcal{A}^\sharp)\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger \hat{Q}^{(t)}(\mathcal{S}^\sharp, a) - Q^*(s, \mathcal{A}^\sharp)\big[Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger Q^*(\mathcal{S}^\sharp, a) \\ &\le \hat{Q}^{(t)}(s, \mathcal{A}^\sharp)\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger \hat{Q}^{(t)}(\mathcal{S}^\sharp, a) - Q^*(s, \mathcal{A}^\sharp)\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger Q^*(\mathcal{S}^\sharp, a) \\ &\quad + Q^*(s, \mathcal{A}^\sharp)\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger Q^*(\mathcal{S}^\sharp, a) - Q^*(s, \mathcal{A}^\sharp)\big[Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger Q^*(\mathcal{S}^\sharp, a) \\ &= \mathrm{Tr}\Big(\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger \cdot \Big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, a)\hat{Q}^{(t)}(s, \mathcal{A}^\sharp) - Q^*(\mathcal{S}^\sharp, a)Q^*(s, \mathcal{A}^\sharp)\Big]\Big) \\ &\quad + \mathrm{Tr}\Big(\Big\{\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger - \big[Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger\Big\} \cdot Q^*(\mathcal{S}^\sharp, a)Q^*(s, \mathcal{A}^\sharp)\Big). \end{aligned}$$

Since $\big|\mathrm{Tr}(AB)\big| \le \sqrt{\text{rank } B}\|A\|_{op}\|B\|_F$, we obtain

$$\begin{aligned} \big|\bar{Q}^{(t)}(s,a) - Q^*(s,a)\big| &\le \sqrt{2}\Big\|\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger\Big\|_{op}\Big\|\hat{Q}^{(t)}(\mathcal{S}^\sharp, a)\hat{Q}^{(t)}(s, \mathcal{A}^\sharp) - Q^*(\mathcal{S}^\sharp, a)Q^*(s, \mathcal{A}^\sharp)\Big\|_F \qquad (3.16) \\ &\quad + \Big\|\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger - \big[Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger\Big\|_{op}\Big\|Q^*(\mathcal{S}^\sharp, a)Q^*(s, \mathcal{A}^\sharp)\Big\|_F. \qquad (3.17) \end{aligned}$$

In the remainder of the proof, we establish upper bounds for (3.16) and (3.17) separately.

- Upper bound for (3.16). Note that $\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp) = Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp) + E$ for some $E \in \mathbb{R}^{|\mathcal{S}^\sharp| \times |\mathcal{A}^\sharp|}$ such that $\|E\|_{\max} \leq \varepsilon$ by assumption. Therefore, $\|E\|_{op} \leq \sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}\|E\|_{\max} \leq \varepsilon\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}$. Since $\sigma_r\big(\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big) \geq \sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big) - \|E\|_{op}$ due to Weyl's inequality, we have

$$\Big\|\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger\Big\|_{op} = \frac{1}{\sigma_r\big(\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)} \leq \frac{1}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big) - \varepsilon\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}} \leq \frac{2}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}$$

provided that $\varepsilon \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)$.

It is easy to see $\Big\|\hat{Q}^{(t)}(\mathcal{S}^\sharp, a)\hat{Q}^{(t)}(s, \mathcal{A}^\sharp) - Q^*(\mathcal{S}^\sharp, a)Q^*(s, \mathcal{A}^\sharp)\Big\|_F \leq \big(2V_{\max}\varepsilon + \varepsilon^2\big)\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}$ because $\big|\hat{Q}^{(t)}(s, a)\hat{Q}^{(t)}(s, a) - Q^*(s, a)Q^*(s, a)\big| \leq 2V_{\max}\varepsilon + \varepsilon^2$ for all $(s, a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$.

- Upper bound for (3.17). We derive an upper bound on $\big\|\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger - \big[Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger\big\|_{op}$ using a classical result on the perturbation of pseudoinverses. By Theorem 3.3 of [153], for any $A$ and $B$ with $B = A + \Delta$,

$$\|B^\dagger - A^\dagger\|_{op} \leq \frac{1 + \sqrt{5}}{2}\max\{\|A^\dagger\|_{op}^2, \|B^\dagger\|_{op}^2\}\|\Delta\|_{op}$$

Therefore,

$$\big\|\big[\hat{Q}^{(t)}(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger - \big[Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big]^\dagger\big\|_{op} \leq \frac{1 + \sqrt{5}}{2} \cdot \frac{4}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)^2} \cdot \varepsilon\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}.$$

Also, it is easy to see that $\big\|Q^*(\mathcal{S}^\sharp, a)Q^*(s, \mathcal{A}^\sharp)\big\|_F \leq V_{\max}\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}$.

All in all, inserting the upper bounds back into (3.16) and (3.17), we have

$$\begin{aligned}\big|\bar{Q}^{(t)}(s, a) - Q^*(s, a)\big| &\leq \frac{2\sqrt{2}}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\big(2V_{\max}\varepsilon + \varepsilon^2\big)\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|} \\ &\quad + \frac{2(1 + \sqrt{5})}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)^2}V_{\max}|\mathcal{S}^\sharp||\mathcal{A}^\sharp|\varepsilon \\ &\leq \left(6\sqrt{2}\left(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\right) + 2(1 + \sqrt{5})\left(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\right)^2\right)V_{\max}\varepsilon.\end{aligned}$$

□

**Proof of Theorem 7.** We take the same approach as in the proof of Theorem 6; the proof is essentially the same as the proof of Theorem 5, with the assumption on the matrix estimation oracle (i.e., Assumption 2) replaced with the explicit guarantee provided in Proposition 3.

As before, the only subtlety comes from that Proposition 3 is a "local" guarantee that holds only for $\varepsilon \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\big)$ whereas Assumption 2 is a global condition that holds for any $\varepsilon$. This requires us to ensure $\max_{(s,a)\in\Omega^{(t)}}\big|\hat{Q}^{(t)}(s,a)-Q^*(s,a)\big| \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\big)$ for all $t=1,\ldots,T$, but the argument in the proof of Theorem 5 itself remains valid.

We make exactly the same choice of algorithmic parameters $\beta^{(t)}, N^{(t)}$ as in the proof of Theorem 5:

$$\beta^{(t)} = \frac{V_{\max}}{8\zeta}(2\gamma\mathsf{c}_{\mathsf{me}})^t \quad \text{and} \quad N^{(t)} = \frac{8}{(2\gamma\mathsf{c}_{\mathsf{me}})^{2(t-1)}}\log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)$$

with an adaptation $\mathsf{c}_{\mathsf{me}} = c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)$ as suggested in Proposition 3. To complete the proof, it suffices to show that $\max_{(s,a)\in\Omega^{(t)}}\big|\hat{Q}^{(t)}(s,a)-Q^*(s,a)\big| \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\big)$ for all $t=1,\ldots,T$.

In the rest of the proof, we prove the above claim by mathematical induction. For $t=0,\ldots,T$, we let $B^{(t)} := \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}}\big|Q^{(t)}(s,a)-Q^*(s,a)\big|$. We can see that if $B^{(t-1)} \leq \big(2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\big)^{t-1}V_{\max}$, then with probability at least $1-\frac{\delta}{T}$, the following two inequalities hold:

1. $\max_{(s,a)\in\Omega^{(t)}}\big|\hat{Q}^{(t)}(s,a)-Q^*(s,a)\big| \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\big)$, and
2. $B^{(t)} \leq 2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma B^{(t-1)}$.

- To prove the first claim, we apply Lemma 10 (see also (3.9)) and observe that with probability at least $1-\frac{\delta}{T}$,

$$\begin{aligned}\max_{(s,a)\in\Omega^{(t)}}\big|\hat{Q}^{(t)}(s,a)-Q^*(s,a)\big| &\leq \gamma\left(B^{(t-1)}+\sqrt{\frac{2V_{\max}^2}{N^{(t)}}\log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)}\right)\\ &\leq \frac{3}{2}\gamma B^{(t-1)} \leq \frac{3}{2}\gamma V_{\max}\\ &\leq \frac{3}{4c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}V_{\max}.\end{aligned}$$

  Here, the last two inequalities follow from the assumptions that $\gamma \leq \frac{1}{2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}$ and that $B^{(t-1)} \leq \big(2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\big)^{t-1}V_{\max} \leq V_{\max}$. Then it suffices to show that

$$\frac{3}{4c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}V_{\max} \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\big). \tag{3.18}$$

  Recall that $c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp) = \Big(6\sqrt{2}\big(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp))}\big)+2(1+\sqrt{5})\big(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp))}\big)^2\Big)V_{\max}$. Also,

we observe that $\sigma_r(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)) > 0$ by definition of the anchor states/actions. Thus, (3.18) is satisfied if the following inequality is true:

$$\left(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\right)\left\{2(1+\sqrt{5})\left(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r\big(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)\big)}\right) + 6\sqrt{2} - \frac{3}{2}\right\} \geq 0.$$

This quadratic inequality is satisfied if and only if $\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp))} \geq 0$ or $\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp))} \leq \frac{3-12\sqrt{2}}{4(1+\sqrt{5})}$. Since $r \geq 1$ and $\sigma_r(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp)) > 0$, we always have $\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r(Q^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp))} \geq 0$ and therefore, the inequality in (3.18) is always true.

- The inequality in the second claim follows from the same argument as in the proof of Theorem 5, cf. (3.7), (3.8) and (3.9).

It remains to certify that $B^{(t)} \leq \big(2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\big)^t V_{\max}$ for $t = 0, \ldots, T-1$. First of all, $Q^{(0)}(s,a) \equiv 0$ by assumption, and hence, $B^{(0)} \leq V_{\max}$. Thus, by the second inequality and the condition on $\gamma$, $B^{(t)} \leq 2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma B^{(t-1)} \leq \ldots \leq \big(2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\big)^t B^{(0)} \leq \big(2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\big)^t V_{\max}$ for all $t = 1, \ldots, T-1$ and the proof is complete. □

## ■ 3.6.3 Rank$(Q^*) \approx r$

Here, we prove results for the approximate rank-$r$ setup in Section 3.6.3. The proof of Proposition 4 is omitted due to its similarity to the proof of Proposition 3 with minor modifications.

**Proof of Theorem 8.** In this proof, we repeat the proof of Theorem 7 with necessary modifications. As before, we must ensure $\max_{(s,a)\in\Omega^{(t)}} \big|\hat{Q}^{(t)}(s,a) - Q_r^*(s,a)\big| \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q_r^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\big)$ for all $t = 1, \ldots, T$ to use Proposition 4. For that purpose, we make exactly the same choice of algorithmic parameters $\beta^{(t)}, N^{(t)}$ as in the proof of Theorem 5, i.e., we let

$$\beta^{(t)} = \frac{V_{\max}}{8\zeta}(2\gamma\mathsf{c}_{\mathsf{me}})^t \quad \text{and} \quad N^{(t)} = \frac{8}{(2\gamma\mathsf{c}_{\mathsf{me}})^{2(t-1)}}\log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)$$

with an adaptation $\mathsf{c}_{\mathsf{me}} = \phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)$ as suggested in Proposition 4, cf. (3.13). To complete the proof, it suffices to show that $\max_{(s,a)\in\Omega^{(t)}} \big|\hat{Q}^{(t)}(s,a) - Q_r^*(s,a)\big| \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r\big(Q_r^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\big)$ for all $t = 1, \ldots, T$.

In the rest of the proof, we prove the above claim by mathematical induction. For notational brevity, we use the shorthand notation $\sigma_r := \sigma_r\big(Q_r^*(\mathcal{S}^\sharp,\mathcal{A}^\sharp)\big)$ and $c_r := \phi_c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)$.

For $t = 0, \ldots, T$, we let $B^{(t)} := \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(t)}(s,a) - Q_r^*(s,a)\right|$. We can see that if

$$B^{(t-1)} \leq \big(2c_r\gamma\big)^{t-1} V_{\max} + (1+c_r\gamma)\xi_r \sum_{i=1}^{t-1} \big(c_r\gamma\big)^{i-1},$$

then with probability at least $1 - \frac{\delta}{T}$, the following two inequalities hold:

1. $\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q_r^*(s,a)\right| \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r$, and
2. $B^{(t)} \leq \big(2c_r\gamma\big)^{t} V_{\max} + (1+c_r\gamma)\xi_r \sum_{i=1}^{t} \big(c_r\gamma\big)^{i-1}$.

- To prove the first claim, we apply Lemma 10 (see also (3.9)) and observe that with probability at least $1 - \frac{\delta}{T}$,

$$\begin{aligned}
\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| &\leq \gamma \left( B^{(t-1)} + \sqrt{\frac{2V_{\max}^2}{N^{(t)}} \log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)} \right) \\
&\leq \frac{3\gamma}{2}(2c_r\gamma)^{t-1} V_{\max} + \gamma(1+c_r\gamma)\xi_r \sum_{i=1}^{t-1} (c_r\gamma)^{i-1} \\
&\leq \frac{3\gamma}{2} V_{\max} + \frac{(1+c_r\gamma)\gamma}{1-c_r\gamma}\xi_r.
\end{aligned}$$

Since $\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q_r^*(s,a)\right| \leq \max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| + \xi_r$, it suffices to show that

$$\frac{3\gamma}{2} V_{\max} + \Big(\frac{(1+c_r\gamma)}{1-c_r\gamma}\gamma + 1\Big)\xi_r \leq \frac{1}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}\sigma_r.$$

Since $\gamma \leq \frac{1}{2c_r} \leq 1$, the above inequality is satisfied if the following inequality is true:

$$\frac{3}{4c_r}\Big(V_{\max} + 2\xi_r\Big) + \xi_r \leq \frac{\sigma_r}{2\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}} \tag{3.19}$$

We observe that $\sigma_r > 0$ by definition of $r$-anchor states/actions, cf. Definition 2. Also, recall from (3.13) that

$$\begin{aligned}
c_r &= \left(6\sqrt{2}\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r} + 2(1+\sqrt{5})\Big(\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r}\Big)^2\right) V_{\max} \\
&\geq 6\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r}\left(1 + \frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r}\right) V_{\max} \\
&> 0.
\end{aligned}$$

Therefore, (3.19) is satisfied if

$$1+2\frac{\xi_r}{V_{\max}} \leq \left(\frac{1}{2}\frac{\sigma_r}{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}-\xi_r\right)8\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r}\left(1+\frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r}\right).$$

We introduce a variable $X = \frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r}$ (which is always positive) to rewrite this inequality as

$$8\xi_r X^2 + (8\xi_r - 4)X - 3 + 2\frac{\xi_r}{V_{\max}} \leq 0.$$

It is easy to see that the above inequality is satisfied if

$$X_- \leq X \leq X_+$$

where $X_\pm = \frac{-(4\xi_r-2)\pm\sqrt{(4\xi_r-2)^2+8\xi_r(3-2\frac{\xi_r}{V_{\max}})}}{8\xi_r}$. As $X_- < 0$ and $X > 0$, we can conclude that (3.19) is satisfied if $X \leq X_+$, which is equivalent to the condition $0 \leq \xi_r \leq \frac{4X+3}{8X^2+8X+\frac{2}{V_{\max}}}$. By assumption, $\xi_r \leq \frac{1}{2X+1+\frac{1}{V_{\max}}}$ and therefore, $\frac{4X+3}{\xi_r} \geq (4X+3)(2X+1+\frac{1}{V_{\max}}) = 8X^2 + (10+\frac{4}{V_{\max}})X + \frac{3}{V_{\max}} + 3 \geq 8X^2 + 8X + \frac{2}{V_{\max}}$. Thus, $0 \leq \xi_r \leq \frac{4X+3}{8X^2+8X+\frac{2}{V_{\max}}}$ is satisfied. Consequently, (3.19) is also satisfied, and the first inequality is proved.

- To prove the second claim, We revisit (3.7), (3.8) and (3.9) in the proof of Theorem 5. Note that nothing has changed for Step 4 (interpolation) and we obtain the same upper bound as in (3.7):

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(t)}(s,a) - Q^*(s,a)\right| \leq \max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}} \left|\bar{Q}^{(t)}(s,a) - Q^*(s,a)\right| + 2\zeta\beta^{(t)}. \quad (3.20)$$

For Step 3 (matrix completion), it follows from Proposition 4:

$$\max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}} \left|\bar{Q}^{(t)}(s,a) - Q^*(s,a)\right| \leq c_r\Big(\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| + \xi_r\Big) + \xi_r. \quad (3.21)$$

The above inequality follows from the observation that

$$\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q_r^*(s,a)\right| \leq \max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| + \xi_r,$$
$$\max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}} \left|\bar{Q}^{(t)}(s,a) - Q^*(s,a)\right| \leq \max_{(s,a)\in\mathcal{S}^{(t)}\times\mathcal{A}^{(t)}} \left|\bar{Q}^{(t)}(s,a) - Q_r^*(s,a)\right| + \xi_r.$$

Lastly, applying Lemma 10 and taking union bound over $(s,a) \in \Omega^{(t)}$, we can show

that

$$\max_{(s,a)\in\Omega^{(t)}} \left|\hat{Q}^{(t)}(s,a) - Q^*(s,a)\right| \le \gamma \left( B^{(t-1)} + \sqrt{\frac{2V_{\max}^2}{N^{(t)}} \log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)} \right) \tag{3.22}$$

with probability at least $1 - \frac{\delta}{T}$.

Combining (3.20), (3.21) and (3.22) yields that the following holds with probability at least $1 - \frac{\delta}{T}$

$$\begin{aligned}
B^{(t)} &\le c_r \left\{ \gamma \left( B^{(t-1)} + \sqrt{\frac{2V_{\max}^2}{N^{(t)}} \log\left(\frac{2|\Omega^{(t)}|T}{\delta}\right)} \right) + \xi_r \right\} + \xi_r + 2\zeta\beta^{(t)} \\
&= c_r\gamma B^{(t-1)} + c_r\gamma(2c_r\gamma)^{t-1}V_{\max} + (c_r\gamma + 1)\xi_r \\
&\le c_r\gamma \left\{ (2c_r\gamma)^{t-1}V_{\max} + (1 + c_r\gamma)\xi_r \sum_{i=1}^{t-1} (c_r\gamma)^{i-1} \right\} + c_r\gamma(2c_r\gamma)^{t-1}V_{\max} + (c_r\gamma + 1)\xi_r \\
&= (2c_r\gamma)^t V_{\max} + (c_r\gamma + 1)\xi_r \sum_{i=1}^{t} (c_r\gamma)^{i-1}.
\end{aligned}$$

It remains to certify that for $t = 0, \ldots, T-1$,

$$B^{(t)} \le (2c_r\gamma)^t V_{\max} + (1 + c_r\gamma)\xi_r \sum_{i=1}^{t} (c_r\gamma)^{i-1},$$

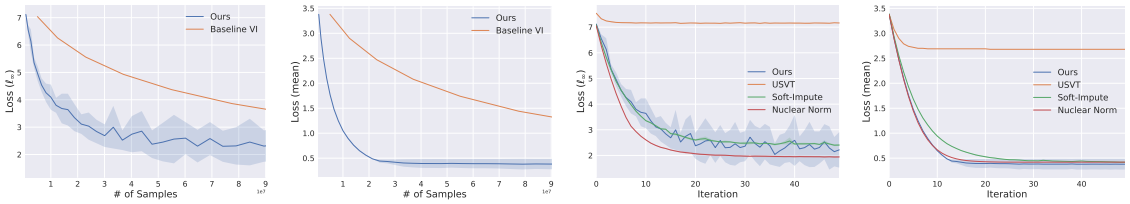
First of all, $Q^{(0)}(s,a) \equiv 0$ by assumption, and hence, $B^{(0)} \le V_{\max}$. Thus, by the second inequality, this condition is satisfied for all $t = 1, \ldots, T-1$ and the proof is complete. □

## ■ 3.7 Empirical Evaluation

Besides theory, we empirically validate the effectiveness of our method on 5 continuous control tasks. The detailed setup can be found in Appendix B.3. In short, we first discretize the spaces into very fine grid and run standard value iteration to obtain a proxy of $Q^*$. The proxy has a very small approximate rank in all tasks; we hence use $r = 10$ for our experiments. As mentioned, we simply select $r$ states and $r$ actions that are far from each other in their respective spaces as our anchor states and actions. For example, if the space is 2-dimensional, we uniformly divide it into $r$ squares and sample one from each square. Because of unavoidable discretization error, we also provide results on mean error, which might be a more reasonable measure in practice. While our proof requires small $\gamma$, we find the method to be generally applicable with large $\gamma$ in real tasks. Therefore, we use

$\gamma = 0.9$ in all the tasks. Additional results on this aspect as well as results on all 5 tasks are provided in Appendix B.4. Below we use the Inverted Pendulum control task to illustrate our performance, but we note that the conclusion remains the same across tasks.

**Improved Sample Complexity with ME.** First, we confirm that the sample complexity of our algorithm improves with the use of ME. Our baseline is the same algorithm described in Section 3.3, but without the ME step (Step 3), i.e., we explore and update all $(s, a) \in \mathcal{S}^{(t)} \times \mathcal{A}^{(t)}$, which is equivalent to performing a simulated value iteration on the discretized set (a.k.a. the synchronous model for $Q$ learning). We illustrate the sample complexity for achieving different levels of $\ell_\infty$ error (Figure 3.2(a)) and mean error (Figure 3.2(b)). It is clear from the plots that our algorithm uses significantly less samples to achieve error at a similar level to the baseline. This evidences that exploiting structure leads to improved efficiency. The same conclusion holds for the other tasks.

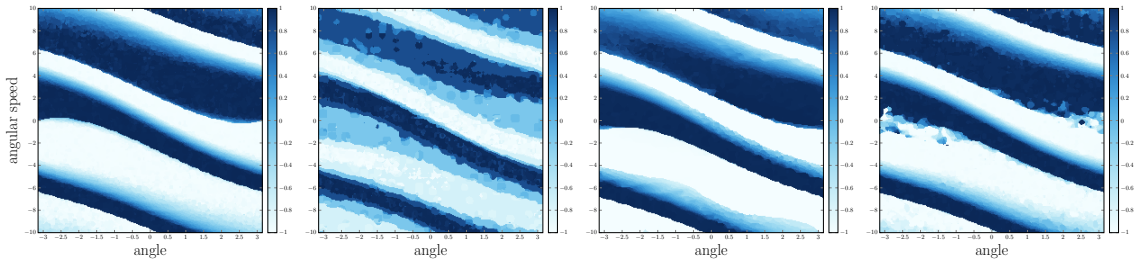



(a) Sample Complexity (b) Sample Complexity (c) $\ell_\infty$ Errors (d) Mean Errors

**Figure 3.2.** Empirical results on the Inverted Pendulum control task. In (a) and (b), we show the improved sample complexity for achieving different levels of $\ell_\infty$ error and mean error, respectively. In (c) and (d), we compare the $\ell_\infty$ error and the mean error for various ME methods. Results are averaged across 5 runs for each method.

**Error Guarantees.** Next, we compare our ME method with other popular ME methods to validate its performance. While theoretically insufficient for RL applications, some established ME methods [31, 34, 118] work well in practice. We compare the methods by feeding the same number of samples of size $O(\max\{\mathcal{S}^{(t)}, \mathcal{A}^{(t)}\})$. As in Figures 3.2(c) and 3.2(d), our method displays a competitive performance, both in $\ell_\infty$ and mean errors. Also, we note that our simple, but powerful method is computationally much more efficient, compared to other methods based on optimization, etc. It can be $40\times$ faster than nuclear norm minimization, cf. Table B.1 in Appendix B.4.2. Overall, these results emphasize the practical value of our method beyond its theoretical soundness. Lastly, we remark that other ME methods also show promise in our experiments; it is certainly a valuable open question to harmonize the established ME methods with low-rank RL.

**Resulting Policy.** As a final proof of concept, in Figure 3.3, we observe that the eventual performance of the policy obtained from the output $Q^{(T)}$ is very close to the policy obtained from $Q^*$. We further summarize the results for standard performance metrics used




(a) Optimal Policy (b) Soft-Impute (c) Nuclear Norm (d) Ours

**Figure 3.3.** Policy visualization of different methods on the Inverted Pendulum control task. The policy is obtained from the output $Q^{(T)}$ by taking $\arg\max_{a\in\mathcal{A}} Q^{(T)}(s,a)$ at each state $s$.

in Table 3.3. For example, the performance metric for Inverted Pendulum is the angular deviation from the target position. Obviously, our efficient method exhibits very competitive performance.

**Table 3.3.** Performance metric for different stochastic control tasks using different ME methods. A.D. stands for angular deviation, T.G. stands for time-to-goal; for both metrics, the smaller the better.

| Method | Optimal | USVT [34] | Soft-Impute [118] | Nuclear Norm [31] | Ours |
|---|---|---|---|---|---|
| Inverted Pendulum (A.D.) | 1.6 ± .0 | 22.5 ± 2.5 | 5.3 ± .6 | 3.1 ± .3 | 3.4 ± .7 |
| Mountain Car (T.G.) | 75.0 ± .3 | 358.8 ± 5.0 | 168.4 ± 8.1 | 92.4 ± 2.8 | 91.8 ± 7.2 |
| Double Integrator (T.G.) | 199.5 ± .1 | 200.0 ± .4 | 199.9 ± .3 | 199.6 ± .2 | 199.7 ± .4 |
| Cart-Pole (A.D.) | 10.1 ± .0 | 19.2 ± 1.0 | 10.4 ± .1 | 10.2 ± .1 | 10.2 ± .2 |
| Acrobot (A.D.) | 2.4 ± .0 | 28.8 ± 4.3 | 9.1 ± 1.2 | 5.1 ± .8 | 6.2 ± 1.0 |

## ■ 3.8 Discussion on Matrix Estimation

Before closing this chapter, we provide discussions on aspects of our ME method to facilitate a more comprehensive understanding. Recall that for the success of our analysis, it is imperative for the matrix estimation subroutine to satisfy Assumption 2. The assumption ensures that the matrix estimation method in use does not amplify the $\ell_\infty$ error too wildly.

**Why Existing Methods Fail.** As reviewed in Related Work, despite the huge success of low-rank matrix estimation, currently available analysis for the existing methods only provides a handle on the estimation error in a few limited class of norms, with no satisfactory recovery guarantees so far on the $\ell_\infty$ error. As a result, we were not able to directly "plug in" existing ME methods and their analysis. We do not believe this is an algorithmic failure of the ME methods, but it is rather a limitation stemming from the disparity between the traditional analysis in ME and the needs in RL application. For example, considering the error in Frobenius norm is natural in the ME tradition for several reasons, but that analysis is not sufficient for applications where entrywise error is more important. Moreover, it

seems manageable, but is not straightforward at once how the mathematical conditions for matrix recovery in ME literature will translate in the context of RL. For example, the finite-dimensional incoherence condition between the principal subspaces and the measurements in ME could translate to a similar infinite-dimensional version of incoherence condition, but some efforts would be needed to reforge existing ME analysis to fit in RL applications seamlessly.

**Why Our ME Method Works.** Instead, we develop an alternative ME subroutine, which is simple, yet sufficiently powerful for our RL task, thereby enabling us to achieve the ultimate conclusion of improved sample complexity. The proposed method is amenable to $\ell_\infty$-error analysis facilitated by matrix algebra. At first glance, our proposal seems extremely simple, and one might doubt its efficacy, e.g., worrying about its numerical stability because it involves the pseudoinverse of a matrix. That concern is partly true, but indeed, there are two key factors that make our method work for the problem of our interest.

First, we assume the existence of "anchor" states and actions, which contain all necessary information for the global recovery of $Q^*$. From a theoretical point of view, this assumption is related to the eigen-gap condition and the incoherence condition between eigenspace and the sampling operator, which are commonly assumed in existing ME literature. From a practical perspective, this means the existence of faithful representatives that reflect the "diversity" of states and actions, which is the case in many real-world applications. Second, crucially, we are not only passively fed with data, but can actively decide which data to collect. Note that our algorithm requires full measurement for the two cylinders (rectangles when represented as a matrix) corresponding to the anchor states and anchor actions without any missing values in them. This is feasible by adaptive sampling, which is not achievable by random sampling. As a byproduct, active sampling allows us to get rid of the spurious log term that appears in sample complexity of existing ME methods as a result of random sampling.

All in all, our ME method is expected to perform reasonably well in the setup considered in this work. We have confirmed this is the case with extensive experiments in Section 3.7.

**Open Questions for Future Work.** Our empirical results evidence that the proposed ME method is successful in the extremely sample deficient-setting where $|\Omega^{(t)}| \asymp \max\{|\mathcal{S}^{(t)}|, |\mathcal{A}^{(t)}|\}$. However, it seems that other existing ME methods based on convex programs also work similarly well, which cannot be explained with the current analysis. As a matter of fact, when the computation cost is ignored, convex-relaxation-based approach is widely accepted as the best one in terms of robustness. This is glimpsed by the evolution of $\ell_\infty$ error in our experiments; unlike the fluctuations observed in our method and soft-impute, the error steadily decreases for the nuclear norm minimization. Also, the existing ME methods might perform better as $|\Omega^{(t)}|$ becomes larger, which is often what traditional ME analysis considers.

Therefore, how to harmonize existing ME methods and the low-rank RL task we consider in this chapter would be an exciting open question. This question might be tackled either by devising new proof techniques to obtain stronger error guarantees for existing ME methods or by improving our decoupled error analysis for RL developed in this chapter. We believe both directions are promising and it would be a valuable contribution to make progress in either one.

## ■ 3.9 Chapter Summary

This chapter is motivated by the soaring demand for data-efficient learning algorithms as reinforcement learning becomes increasingly popular in practice. Through the novel lens of low-rank representation of $Q$-function, this work proposes a theoretical framework to devise efficient RL algorithms. The resulting "low-rank" algorithm, which utilizes a novel matrix estimation method, offers both strong theoretical guarantees and appealing empirical performance.

We remark that there remain several interesting open questions. First of all, we believe it is possible to refine the error analysis in this work to achieve a stronger theoretical guarantee. For example, the condition $\gamma < \frac{1}{2\mathsf{c}_{\mathsf{me}}}$ for convergence in Theorem 5 is probably an artifact of our decoupled analysis and is possibly removable. Perhaps, devising better ME methods for the purpose of RL can be a solution to lift the restriction on the range of $\gamma$, which is an interesting problem on its own. Another open question is on identifying sufficient conditions for MDPs that indeed lead to a low-rank $Q^*$. For example, what properties on the transition kernel and reward function might result in the low-rank structure. Also, we conjecture that our "sample and pseudo-explore (via ME)" scheme is more broadly applicable beyond the generative setup considered in this work, e.g., to the setup where one has to follow some policies to explore sequentially and collect trajectories. The most prominent challenge anticipated is that we are no longer able to sample "any" state-action pair completely freely and adaptively; the sampling needs to respect the exploration policy. This difficulty may be overcome with a more refined ME method, with ad-hoc techniques. More broadly, the main insight we develop in this chapter remains valid and applicable to various problems in machine learning beyond RL, which involves learning a bivariate function possessing a low-rank structure. Overall, we believe this chapter can serve as a starting point for fruitful future research along those promising directions.

Chapter 4

# Stability in Unbounded State Space

We continue our journey on designing data-efficient reinforcement learning algorithms, but focusing on unbounded domains in this chapter. In particular, we consider the problem of RL with unbounded state space motivated by the classical problem of scheduling in a queueing network. As previewed in the Introduction (Section 1.2.3), unlike the bounded domains studied in Chapters 2 and 3, there are unique conceptual challenges in this case which must be addressed first in order to design any meaningful learning methods. Recall that traditional policies as well as error metric that are designed for finite, bounded or compact state space, would require infinite samples for providing any meaningful performance guarantee (e.g. $\ell_\infty$ error) for unbounded state space. Approaches that rely on offline training only are bound to fail as system will reach a state that is not observed in and possibly far from the finitely many samples during offline training and hence, there is no meaningful guidance from the policy. As such, to learn a reasonable policy with an unbounded state space, we ought to consider online policy. Furthermore, in analyzing or quantifying "goodness" of such a policy, since the state space is unbounded, expecting a uniformly good approximation of the optimal value function over the entire state space is not a meaningful measure. That is, we need a new notion of performance metric.

As the main contribution of this chapter, inspired by the literature in queueing systems and control theory, we propose *stability* as the notion of "goodness": the state dynamics under the policy should remain in a *bounded* region with high probability. We consider systems that have such a stability structure: the system dynamics under the optimal policy is stable and hence respects a Lyapunov function. As a proof of concept, we propose a simple online RL policy using Sparse-Sampling-based Monte Carlo oracle and argue that it satisfies the stability property. We note that the structural assumption of existence of a Lyapunov function is not restrictive as it is equivalent to the positive recurrence or stability property of any Markov chain, i.e., if there is any policy that can stabilize the system then it must possess a Lyapunov function. And importantly, our policy does not utilize the knowledge of the specific Lyapunov function.

While stable, the naive policy is sample-inefficient in regimes of interest. Recall that

earlier in the chapter summary of Chapter 2 (Section 2.10), we mentioned data reuse as an important practical "trick" for achieving better performance. To make our method sample efficient, we provide an improved, sample efficient Sparse-Sampling-based Monte Carlo oracle with Lipschitz value function that may be of interest in its own right. The oracle utilizes a minimal structure in the value function, the Lipschitz smoothness, to replace newly encountered state during Monte Carlo simulation by its closest state in a pre-fixed subset of states. As such, the transition data for the fixed subset of states may be used multiple times during the simulations.

Furthermore, while the above RL policy does not require knowing the Lyapunov function itself, it does have a parameter whose optimal value depends on the unknown drift parameter of the Lyapunov function. To address this issue, we design an adaptive version of the algorithm, based on carefully constructed statistical tests, which discovers the correct tuning parameter automatically. Finally, since this chapter is primarily motivated by the classical problem of queueing networks, we remark that in the context of a queueing network, the performance guarantee of stability for general RL algorithm would imply the so-called "rate stability" of the queueing network. We will elaborate with examples in the coming sections.

**Organization of Chapter 4.** Chapter 4 is structured as follows. Section 4.1 starts with a short literature review comparing RL safety and stability. Section 4.2 then introduces the framework and formally defines the notion of stability considered in this chapter. We will also provide queueing examples to understand the conditions involved. In Section 4.3, we describe three online algorithms and state their stability guarantees: (1) Section 4.3.1 presents a simple stable policy utilizing a Sparse Sampling Monte Carlo oracle; (2) Section 4.3.2 improves its data efficiency by exploiting the Lipschitz structure; (3) Section 4.3.3 further proposes an adaptive algorithm for tuning the parameter, and this completes our study on stability. We also discuss the implication of our results in the context of the queueing examples. The proofs of our main theorems are provided in Sections 4.4–4.6. We conclude in Section 4.7 with discussions of future directions.

## ■ 4.1 Related Work

The concept of stability introduced in this chapter is related to the notion of *safety* in RL, but with crucial differences. Various definitions of safety exist in the literature [130, 63]. One line of work defines safety as hard constraints on individual states and/or actions [62, 73, 48, 50, 96]. Some other work considers safety guarantee in terms of keeping the expected sum of certain costs over a trajectory below a given threshold [2, 40, 182]. In our work, stability is defined in a way akin to the positive recurrence of Markov chains, which cannot be immediately written as constraints/costs over the states and actions. In particular, our

**Table 4.1.** Comparison with selected prior work on RL safety and stability.

| | Settings/Conditions | Guarantees |
|---|---|---|
| Ours | Unbounded space; Unknown dynamics; Existence of unknown Lyapunov function | Stochastic stability |
| [50] | Linear dynamics; Unknown parameters; Quadratic cost (LQR) | Constraints on state & action |
| [165] | Finite space; Deterministic, known dynamics; Gaussian safety function | Constraints on state & action |
| [182] | Unknown dynamics; Compact parametrized policy class | Expected constraint costs (CMDP) |
| [168] | Gaussian process dynamics; Unknown parameters | Control-theoretic stability |
| [20] | Compact space; Deterministic, partially known dynamics; Access to Lyapunov func. | Control-theoretic stability |

stability notion captures long-term behavior of the system — it should eventually stay in a desirable region of the state space with high probability. In general, there does not exist an action that immediately drives the system back to that region; learning a policy that achieves so in the long run is non-trivial and is precisely our goal. Overall, we believe this notion of stability provides a generic, formal framework for studying RL with unbounded state space.

Many work on RL safety is model-based, either requiring a prior known safe backup policy [62, 73], or using model predictive control approaches [169, 135, 11, 96]. One line of work focuses specifically on systems with a linear model with constraints (e.g., LQR) [36, 50]. Some other work considers model-free policy search algorithms [2, 40, 182], under the framework of constrained Markov Decision Process (CMDP) [7], which models safety as expected cumulative constraint costs.

Another line of work considers control-theoretic notions of stability [19, 168, 20], which bear similarity to our framework. We remark that these results mostly focus on systems with deterministic and partially unknown dynamics, different from our setting where the dynamics are stochastic and unknown. Their approaches are limited to compact state spaces where discretization is feasible.

Our analysis makes use of Lyapunov function, which is a classical tool in control and Markov chain theory for studying stability and steady-state behaviors [152, 65, 57]. The work by [131] is among the first to use Lyapunov functions in RL and studies closed-loop stability of an agent. More recent work uses Lyapunov functions to establish the finite-time error bounds of TD-learning [151], to solve constrained MDPs [40] and to find region of attraction for deterministic systems [20, 19].

Our RL algorithm fits broadly into value-based methods [175, 157, 122, 167, 180, 143]. In terms of queueing networks, approximate dynamic programming techniques and RL have been applied in prior work [99, 134, 123], though their settings and goals are quite different from us, and their approaches exploit prior knowledge of queueing theory and specific structures of the problems. Most related to us is the recent work [111], which also considers problems with unbounded state space. Their algorithm, however, makes use of a *known*

stabilizing policy. We do not assume knowledge of such a policy; rather, our goal is to learn one from data.

## 4.2 Setup and Notion of Stability

In this section, we formally describe the setup and problem statement. We first state the class of MDPs studied in this chapter in Section 4.2.1. Then, we introduce our notion of stability in Section 4.2.2 and provide some related discussions.

### 4.2.1 Markov Decision Process and Online Policy

**Markov Decision Process (MDP).** As in Chapters 2 and 3, we consider the usual setup of discounted Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. For simplicity, we consider a deterministic reward function and use the notation $R(s,a)$ as in Chapter 3. At time $t$, the action $a_t$ is chosen as per some policy $\pi^t \equiv [\pi^t(a|s),\ a \in \mathcal{A}, s \in \mathcal{S}]$, where $\pi^t(a|s)$ represents the probability of taking action $a_t = a$ given state $s_t = s$. If $\pi^t = \pi$ for all $t$, then it is called a stationary policy. Recall the definitions of value function and $Q$-function in Section 1.1. As we consider unbounded state space in this chapter, in addition to the generic setup, we focus on MDPs satisfying the following condition.

**Condition 1.** *Action space $\mathcal{A}$ is finite, and state space $\mathcal{S} \subseteq \mathbb{R}^d$ is unbounded with some $d \geq 1$.*

We assume that the reward function $R$ is bounded and takes value in $[0, R_{\max}]$. Consequently, for *any* policy, $V$ and $Q$ functions are bounded and take value in $[0, V_{\max}]$, where $V_{\max} \triangleq R_{\max}/(1-\gamma)$. Breaking ties randomly, we shall restrict to optimal policy $\pi^*$ of the following form: for each $s \in \mathcal{S}$,

$$\pi^*(a|s) = \begin{cases} \frac{1}{|\mathcal{A}^*(s)|} & \text{if } a \in \mathcal{A}^*(s), \\ 0 & \text{otherwise,} \end{cases}$$

where $\mathcal{A}^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s,a)$. Define $\Delta_{\min}(s) = \max_{a \in \mathcal{A}} Q^*(s,a) - \max_{a \in \mathcal{A} \setminus \mathcal{A}^*(s)} Q^*(s,a)$, i.e., it is the gap between $Q^*$ and the value of the second optimal action for state $s$. For a given state $s \in \mathcal{S}$, if not all actions are optimal, then $\Delta_{\min}(s) > 0$. Denote the minimum gap by $\Delta_{\min} \triangleq \inf_{s \in \mathcal{S}} \Delta_{\min}(s)$. Throughout the chapter, we assume that $\Delta_{\min} > 0$. Finally, like Chapters 2 and 3, we assume the access to a generative model so that simulation of the underlying system is possible at each step.

**System Dynamics and Online Policy.** As discussed, our interest in this chapter is in designing an online policy starting with no prior information. Precisely, the system

starts with arbitrary initial state $s_0 \in \mathcal{S}$ and initial policy $\pi^0$. At time $t \geq 0$, given state $s_t \in \mathcal{S}$, action $a_t \in \mathcal{A}$ is chosen with probability $\pi^t(a_t|s_t)$ which leads to state $s_{t+1} \in \mathcal{S}$ with probability $\mathcal{P}(s_{t+1}|s_t, a_t)$. At each time $t$, policy $\pi^t$ is decided using potentially a finite number of simulation steps from the underlying MDP, in addition to the historical information observed till time $t$. In this sense, the policy is *online*.

### ■ 4.2.2 Stability

We desire our online policy to have a *stability* property, formally defined as follows.

**Definition 3** (Stability). *We call policy $\{\pi^t,\ t \geq 0\}$ stable if for any $\theta \in (0,1)$, there exists a bounded set $\mathcal{S}(\theta) \subset \mathcal{S}$ such that the following are satisfied:*

1. *Boundedness:*
$$\limsup_{t\to\infty} \mathbb{P}\big(s_t \notin \mathcal{S}(\theta) \mid s_0 = s\big) \leq \theta, \quad \forall\, s \in \mathcal{S}. \tag{4.1}$$
2. *Recurrence: Let $T(s,t,\theta) = \inf\{k \geq 0 : s_{t+k} \in \mathcal{S}(\theta) \mid s_t = s\}$.[1] Then,*
$$\sup_{t\geq 0} \mathbb{E}\big[T(s,t,\theta)|s_t = s\big] < \infty, \quad \forall\, s \in \mathcal{S}. \tag{4.2}$$

In words, we desire that starting with *no prior information*, the policy learns as it goes and manages to retain the state in a finite, bounded set with high probability. It is worth remarking that the above stability property is similar to the recurrence property for Markov chains.

**Problem Statement.** With the definition of stability, our formal goal is hence to design an *online* stable policy for a given, discounted MDP.

**MDPs Respecting Lyapunov Function.** As we search for a stable policy for MDP, if the optimal stationary policy for MDP is stable, then the resulting system dynamics under the optimal policy is a time homogeneous Markov chain that is positive recurrent. The property of positive recurrence is known to be equivalent to the existence of the so-called *Lyapunov* function; see [120]. In particular, if there exists a policy for the MDP under which the resulting Markov chain is positive recurrent, then this policy has a Lyapunov function. These observations motivate us to restrict attention to MDPs with the following property.

**Condition 2.** *The Markov chain over $\mathcal{S}$ induced by MDP operating under the optimal policy $\pi^*$ respects a Lyapunov function $L : \mathcal{S} \to \mathbb{R}_+$ such that $\liminf_{\|s\|\to\infty} L(s) = \infty$, $\sup\big\{\|s\| : L(s) \leq \ell\big\} < \infty$ for all finite $\ell$, and such that for some $\nu, \nu', B, \alpha > 0$, for any $t \geq 0$:*

[1] inf of empty set is defined as $\infty$.

1. *(Bounded Increment) For every $s_t \in \mathcal{S}$ and every $a \in \mathcal{A}$,*

$$|L(s_{t+1}) - L(s_t)| \leq \nu, \quad \|s_{t+1} - s_t\| \leq \nu', \tag{4.3}$$

*for all possible next states $s_{t+1}$ with $\mathcal{P}(s_{t+1}|s_t, a) > 0$.*[2]

2. *(Drift Condition) For every $s \in \mathcal{S}$ such that $L(s) > B$,*

$$\mathbb{E}\big[L(s_{t+1}) - L(s_t)|s_t = s\big] \leq -\alpha. \tag{4.4}$$

As explained above, the requirement of MDP respecting Lyapunov function under the optimal policy is not restrictive. Recent work by [47] also considers a similar Lyapunov condition for analyzing the proximal policy optimization algorithm [141]. We further note that our policy *does not* require *precise* knowledge of such a Lyapunov function.

**Queueing Examples.** Since we have been using queueing systems to motivate our investigation on unbounded state space, it would be informative to discuss examples that indeed satisfy the conditions from a queueing perspective instead of abstract properties of MDPs. We provide two simple examples below:

1. The first example is a discrete-time single-queue single-server system. Jobs arrive according to a stationary process with rate $\lambda$. At each time $t$, the server can decide to serve or not serve. If it is serving a job, the probability that this job is completed at the end of the time step is $\mu$. The system state can be represented by the queue length, which can be unbounded. Let the reward at each time be the negative change of the queue length. With bounded arrivals at each time step, the reward is bounded. It is clear that regardless of the discount factor $\gamma$, the optimal policy is to always serve a job when the queue is non-empty (it is optimal for average reward as well). When $\lambda/\mu < 1$, it is well known that the policy is stable from a queueing system perspective (also known as throughput optimal), i.e., the MDP under the policy is positive recurrent, which implies that the system satisfies Condition 2.

2. The above toy example extends to the multi-queue single-server setting, with arrival rate $\lambda_k$ for queue $k \in [K]$, and service distribution Bernoulli($\mu_k$) for jobs from queue $k$. At each time step, the scheduling decision for the server is which queue to serve. Let the reward at each time be the weighted sum of the negative change of each queue, with weight $c_k$ for queue $k$. The classical $c\mu$ rule that assigns the server to queue $i$ if $c_i\mu_i = \max_{k\in[K]}\{c_k\mu_k|q_k(t) > 0\}$ is optimal, for both discounted reward and average

[2] $\mathcal{P}(\cdot|s_t, a)$ should be understood as the density of the conditional distribution of the next state with respect to the Lebesgue measure (for continuous state space) or the counting measure (for discrete state space).

reward settings, when the service discipline is preemptive [28]. It is well known that $c\mu$ rule is stable, whenever the arrival rates $\{\lambda_k\}_{k\in[K]}$ satisfy $\sum_{k\in[K]} \lambda_k/\mu_k < 1$. In particular, it can be verified that for the $c\mu$ rule, $L(q) = \sum_{k\in[K]} \frac{q_k}{\mu_k}$ serves as a Lyapunov function satisfying Condition 2.

While simple, these examples demonstrate how queueing systems can be naturally connected to our MDP framework. For more sophisticated queueing systems, we note that the optimal queueing policy is often unknown and there has been decades of research in the queueing community focusing on designing policies that stabilize the system. Since these potentially sub-optimal policies are stable, it is reasonable to expect that the optimal policy ought to be stable and hence satisfies Condition 2. After we present the main technical results for our framework, in Sections 4.3.3 and 4.3.4 , we shall re-visit the queueing setting to understand the implication of our framework when specialized to the context of queueing systems.

**Why Stability Matters.** It should be clear now that stability is a much desired property. In many applications, it is convenient to think of maximizing *reward* equivalently as minimizing *cost*. For settings with unbounded state space, states in a small bounded region are usually associated with low cost. Intuitively, stability implies achieving a bounded cost; the classical queueing systems above provide such example applications. An unstable policy leads to unbounded queue lengths: jobs accumulate and experience extremely long or even unbounded delay. For such applications, stability provides a reasonable notion of first-order optimality for cost minimization and hence, achieving stability is a highly desirable *first* goal — just like *rate stability* in queueing systems. Indeed, the next set of goals would be further refined notions of optimality just as the way diffusion or heavy traffic approximations in the queueing systems have played a role.

## ■ 4.3 Online Stable Policy

We present our main results in this section. First, we describe a stationary, online policy that is stable under Conditions 1 and 2. This policy is simple and provides key intuition behind our approach, though being sample inefficient. Next, we present an efficient version thereof that utilizes the minimal structure of Lipschitz optimal value function (cf. Condition 3). Finally, we design an adaptive version of our algorithm that automatically discovers the appropriate tuning parameter without knowing the value of the drift parameter of the Lyapunov function.

### ■ 4.3.1 Sample Inefficient Stable Policy

**Overview of Policy.** At each time $t \geq 0$, given the state $s_t \in \mathcal{S}$, an action $a_t \in \mathcal{A}$ is chosen by sampling from a distribution over $\mathcal{A}$. The distribution is determined by an online planning algorithm that uses finitely many simulations of MDP performed at time $t$ and depends only on the input state $s_t$. Therefore, the policy is stationary. Precisely, using Monte Carlo oracle with Sparse Sampling for parameters $C, H > 0$ (details below, adapted from [88]), we produce $\widehat{Q}(s_t, a)$ as an estimate of $Q^*(s_t, a)$, for each action $a \in \mathcal{A}$. We use Boltzman distribution with parameter $\tau > 0$ for sampling action $a_t$:

$$\widehat{\pi}(a|s_t) = \frac{e^{\widehat{Q}(s_t,a)/\tau}}{\sum_{a'} e^{\widehat{Q}(s_t,a')/\tau}}, \forall\, a \in \mathcal{A}.$$

**Sparse Sampling Monte Carlo Oracle.** We describe the Monte Carlo oracle based on sparse sampling [88]. Given an input state $s \in \mathcal{S}$ with two integer valued parameters, $C, H > 0$, and an estimate of value function $\widehat{V}$, it produces an estimate of $Q^*(s, a)$ for all $a \in \mathcal{A}$. The error in the estimate depends on $C, H$ and error in value function $\widehat{V}$. With larger values of $C$ and $H$, the estimation error decreases but the number of simulations of MDP required increases. Next we provide details of the algorithm.

Sparse sampling oracle constructs a tree of depth $H$ representing a partial $H$-step look-ahead of MDP, starting with the input state $s_0 = s$ as its root. The tree is constructed iteratively as follows: at any state (node) $s'$ in the tree (initially, root node $s_0$), for each action $a \in \mathcal{A}$, sample $C$ times the next state of MDP for the state-action pair $(s', a)$. Each of the resulting next states, $C$ for each $a \in \mathcal{A}$, are placed as children nodes, with the edge between $s'$ and the child node labeled by the generated reward. The process is repeated for all nodes of each level until reaching a depth of $H$. That is, it builds an $|\mathcal{A}|C$-array tree of depth $H$ representing a partial $H$-step look-ahead tree starting from the queried state $s_0$, and hence the term *sparse sampling* oracle. Figure 4.1 provides a graphical illustration of the above process.

To obtain an estimate for the $Q$-values associated with $(s_0, a)$, $a \in \mathcal{A}$, we start by assigning value 0 to each leaf node $s_{\mathsf{leaf}}$ at depth $H$, i.e., $\widehat{V}^{(H)}(s_{\mathsf{leaf}}) = 0$. These values, together with the associated rewards on the edges, are then backed-up to find estimates of $Q$-values for their parents, i.e., nodes at depth $H - 1$. The estimate for the $Q$-value of a parent node $s$ and an action $a$ is just a simple average over $a$'s children, i.e., $\widehat{Q}^{(H-1)}(s, a) = R(s, a) + \gamma \frac{1}{C} \sum_{s' \in S_{\mathsf{next}}(s,a,C)} \widehat{V}^{(H)}(s')$, where $R(s, a)$ is the reward on the edge of action $a$, and $S_{\mathsf{next}}(s, a, C)$ is the set of $a$'s children nodes. The estimate for the value of state $s$ is given by $\widehat{V}^{(H-1)}(s) = \max_a \widehat{Q}^{(H-1)}(s, a)$. The process is recursively applied from the leaves up to the root level to find estimates of $\widehat{Q}^{(0)}(s_0, a)$ for the root node $s_0$.

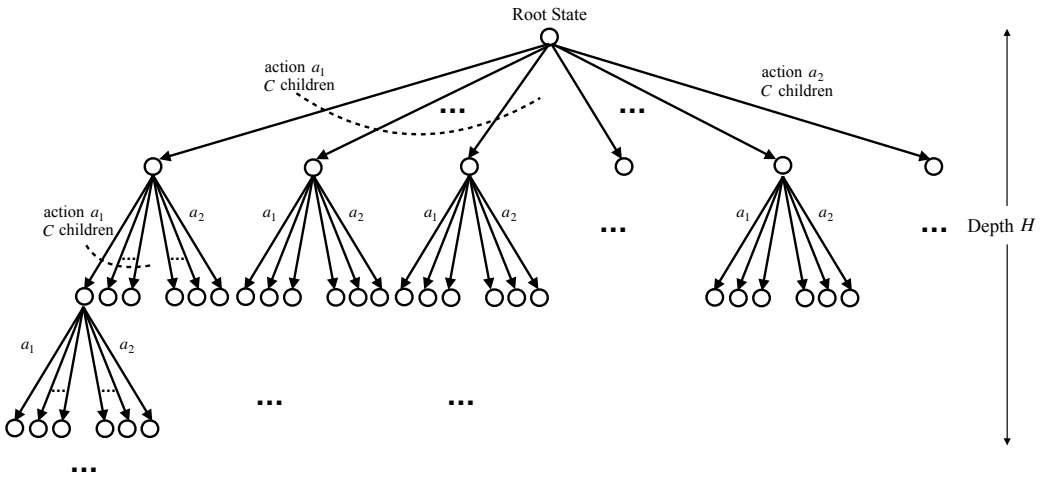



**Figure 4.1.** Illustration of the Monte Carlo oracle with Sparse Sampling. The figure is adapted from [88].

**Lemma 12** (Oracle Guarantee)**.** *Given input state $s \in \mathcal{S}$ and $\delta > 0$, under the Sparse Sampling Monte Carlo Oracle, we have*

$$\max_{a \in \mathcal{A}} |Q^*(s,a) - \widehat{Q}^{(0)}(s,a)| \leq 2V_{\max}\delta, \tag{4.5}$$

*with probability at least $1-\delta$, with choice of parameters*

$$H = \lceil \log_\gamma(\delta) \rceil, \qquad C = \frac{2\gamma^2}{\delta^2(1-\gamma)^2}\Big(2H \log \frac{2\gamma^2|\mathcal{A}|H}{\delta^2(1-\gamma)^2} + \log\frac{2}{\delta}\Big). \tag{4.6}$$

*The number of simulation steps of MDP utilized is $O\big((|\mathcal{A}|C)^H\big)$, which as a function of $\delta$ scales as $O\Big(\big(\frac{1}{\delta^2}\log^2\frac{1}{\delta}\big)^{O(\log\frac{1}{\delta})}\Big)$.*

We shall omit the proof of Lemma 12 as it follows directly from results in [88]. Further, we provide proof of Lemma 13 which establishes similar guarantees for a modified sample-efficient Sparse Sampling Monte Carlo oracle. Finally, we remark that in principle, other Monte Carlo oracle, such as MCTS as introduced in Chapter 2 can be similarly applied here. Indeed, the sparse sampling oracle can be viewed as a simplified MCTS where each action is just uniformly tried $C$ times to build the simulation tree. We choose sparse sampling to convey the intuition because of its algorithmic simplicity and its cleaner dependence on parameters.

**Stability Guarantees.** We state the result establishing stability of the stationary policy described above under appropriate choice of parameters $\delta, \tau, C$ and $H > 0$. The proof is provided in Section 4.4.

**Theorem 9.** *Suppose that the MDP of interest satisfies Conditions 1 and 2. Then, with*

$\delta = \tau^2$, $\tau$ *as*

$$\tau \leq \tau(\alpha) \triangleq \min\left\{\sqrt{\frac{\alpha}{24\nu}}, \frac{1-\gamma}{8R_{\max}}, \frac{(1-\gamma)\alpha}{48R_{\max}|\mathcal{A}|^2\nu}, \frac{\Delta_{\min}}{\ln\left(\frac{24\nu|\mathcal{A}|}{\alpha}\right)}\right\}, \tag{4.7}$$

*and* $C, H > 0$ *chosen as per Lemma 12 for a given* $\delta$*, the resulting stationary policy is stable.*

### ■ 4.3.2 Sample Efficient Stable Policy

Theorem 9 suggests that, as a function of $\alpha$, $\delta(= \tau^2)$ scales as $\min(\alpha, \alpha^2)$. Indeed, as $\alpha$ becomes larger, i.e., the negative drift for Lyapunov function under optimal policy increases, system starts living in a smaller region with a higher likelihood. The challenging regime is when $\alpha$ is small, formally $\alpha \to 0$. Back to our queueing examples, this corresponds to what is known as *Heavy Traffic* regime in queueing systems. Analyzing complex queueing systems in this regime allows understanding of fundamental "performance bottlenecks" within the system and subsequently understanding the properties of desired optimal policy. Indeed, a great deal of progress has been made over more than past four decades now; see for example [74, 72, 103]. Back to the setting of MDP with $\alpha \to 0$: based on Theorem 9, the number of samples required per time step for the policy described in Section 4.3.1 scales as $O\left(\left(\frac{1}{\alpha^4}\log^2\frac{1}{\alpha}\right)^{O(\log\frac{1}{\alpha})}\right)$. That is, the sample complexity of the policy is *super-polynomial* in $1/\alpha$.

**Minimal Structural Assumption.** In what follows, we describe a stable policy with the number of samples required scaling *polynomially* in $1/\alpha$, precisely $\widetilde{O}(1/\alpha^{2d+4})$ where $d$ is the dimension of the state space $\mathcal{S}$. This efficiency is achieved due to minimal structure in the optimal value function in terms of Condition 3, which effectively states that the value function is Lipschitz. Specifically, we provide an efficient Sparse Sampling Monte Carlo oracle exploiting the Bounded Increment property (4.3) in Condition 2 along with Lipschitzness of optimal value function as described in Condition 3. We remark that as in Chapters 2 and 3, for learning with continuous state/action space, smoothness assumption such as the Lipschitz continuity is natural and standard [9, 143, 55, 125].

**Condition 3.** *Let* $\mathcal{S} = \mathbb{R}^d$*. The optimal value function* $V^* : \mathcal{S} \to \mathbb{R}$ *is a Lipschitz continuous function. Precisely, there exists constant* $\zeta > 0$ *such that for any* $s, s' \in \mathcal{S}$*,*

$$\left|V^*(s) - V^*(s')\right| \leq \zeta\|s - s'\|.$$

**Overview of Policy.** The policy is exactly the same as that described in Section 4.3.1, but with a single difference: instead of using the vanilla Sparse Sampling Monte Carlo oracle, we replace it with an efficient version that exploits Condition 3 as described next.

**Efficient Sparse Sampling Monte Carlo Oracle.** We describe a modification of the Monte Carlo oracle based on sparse sampling described earlier. As before, our interest is in obtaining an approximate estimate of $Q^*(s,a)$ for a given $s \in \mathcal{S}$ and any $a \in \mathcal{A}$ with minimal number of samples of the underlying MDP. To that end, we shall utilize property (4.3) of Condition 2 and Condition 3 to propose a modification of the method described in Section 4.3.1. For a given parameter $\varepsilon > 0$, define

$$\mathcal{S}_\varepsilon \triangleq \big\{(k_1\varepsilon, \ldots, k_d\varepsilon) : k_1, \ldots, k_d \in \mathbb{Z}\big\}.$$

For any $s \in \mathcal{S} = \mathbb{R}^d$, there $\exists\, \tilde{s} \in \mathcal{S}_\varepsilon$ so that $\|s - \tilde{s}\|_\infty \leq \varepsilon$ or $\|s - \tilde{s}\| \leq \sqrt{d}\|s - \tilde{s}\|_\infty \leq \varepsilon\sqrt{d}$. Define $S_\varepsilon : \mathcal{S} \to \mathcal{S}_\varepsilon$ that maps $s \in \mathcal{S}$ to its closest element in $\mathcal{S}_\varepsilon$: i.e. $S_\varepsilon(s) \in \arg\min\big\{\|s-\tilde{s}\| : \tilde{s} \in \mathcal{S}_\varepsilon\big\}$.

For a given $s \in \mathcal{S}$, we shall obtain a good estimate of $Q^*(s,a)$ effectively using method described in Section 4.3.1. We start with $s$ as the root node. For each state-action pair $(s,a)$, $a \in \mathcal{A}$, we sample next state of MDP $C$ times leading to states $S_{\mathsf{next}}(s,a,C) \subset \mathcal{S}$. In contrast to the method described in Section 4.3.1, we use states $\{S_\varepsilon(s') : s' \in S_{\mathsf{next}}(s,a,C)\}$ in place of $S_{\mathsf{next}}(s,a,C)$. These form states (or nodes) as part of the sampling tree at level 1. For each state, say $\tilde{s}$ at level 1, for each $a \in \mathcal{A}$, we sample $C$ next states of MDP, and replace them by their closest elements in $\mathcal{S}_\varepsilon$ to obtain states or nodes at level 2; we repeat till $H$ levels. Note that all states on level 1 to $H$ are from $\mathcal{S}_\varepsilon$. To improve sample efficiency, during the construction, if a state $\tilde{s}$ has been visited before, we use previously sampled next states $S_{\mathsf{next}}(\tilde{s},a,C)$ for each action $a \in \mathcal{A}$, instead of obtaining new samples.

To obtain an estimate $\widehat{Q}^{(0)}(s,a)$ for the $Q$-value associated with the root state $s$ and any action $a \in \mathcal{A}$, we start by assigning the value 0 to each leaf node $s_{\mathsf{leaf}}$ at depth $H$, i.e., $\widehat{V}^{(H)}(s_{\mathsf{leaf}}) = 0$. These values, together with the associated rewards on the edges, are then backed-up to find estimates for $Q$-values of their parents at depth $H-1$, and this is repeated till we reach the root node, $s$. Precisely, for $1 \leq h \leq H$ and node $\tilde{s}$ at level $h-1$,

$$\widehat{Q}^{(h-1)}(\tilde{s},a) = R(\tilde{s},a) + \gamma\frac{1}{C}\sum_{s' \in S_{\mathsf{next}}(\tilde{s},a,C)} \widehat{V}^{(h)}\big(S_\varepsilon(s')\big), \qquad \widehat{V}^{(h-1)}(\tilde{s}) = \max_a \widehat{Q}^{(h-1)}(\tilde{s},a). \tag{4.8}$$

The method outputs $\widehat{Q}^{(0)}(s,a)$ as an estimate of $Q^*(s,a)$ for all $a \in \mathcal{A}$.

**Stability Guarantees with Improved Sample Complexity.** In Lemma 13, we summarize the estimation error as well as the number of samples utilized by this modified Sparse Sampling Monte Carlo oracle. The proof is provided in Section 4.5.

**Lemma 13** (Modified Oracle Guarantee)**.** *Given input state $s \in \mathcal{S}$, $\delta > 0$, under the*

*modified Sparse Sampling Monte Carlo oracle, we have*

$$\max_{a\in\mathcal{A}} \left|Q^*(s,a) - \widehat{Q}^{(0)}(s,a)\right| \leq 2V_{\max}\delta,$$

*with probability at least* $1-\delta$*, with choice of parameters* $H = \lceil \log_\gamma(\delta)\rceil$*,* $\varepsilon = \frac{\delta V_{\max}(1-\gamma)}{2\zeta\gamma\sqrt{d}}$*, and*

$$C = \Omega\left(\frac{\gamma^2}{(1-\gamma)^2\delta^2}\left(\log\frac{2|\mathcal{A}|}{\delta} + d\log H + d\log\frac{1}{\varepsilon}\right)\right) \;=\; \Omega\left(\frac{1}{\delta^2}\log\frac{1}{\delta}\right).$$

*The number of simulation steps of MDP utilized, as a function of* $\delta$ *scales as* $O\left(\frac{1}{\delta^{2+d}}\log^{1+d}\frac{1}{\delta}\right)$.

Next, we state the result establishing stability of the stationary policy described above under appropriate choice of parameters $\delta, \tau, C$ and $H > 0$.

**Theorem 10.** *Let the MDP of interest satisfy Conditions 1, 2 and 3. Then, with* $\delta = \tau^2$*,* $\tau$ *as*

$$\tau \leq \tau(\alpha) \triangleq \min\left\{\sqrt{\frac{\alpha}{24\nu}}, \frac{1-\gamma}{8R_{\max}}, \frac{(1-\gamma)\alpha}{48R_{\max}|\mathcal{A}|^2\nu}, \frac{\Delta_{\min}}{\ln\left(\frac{24\nu|\mathcal{A}|}{\alpha}\right)}\right\}, \tag{4.9}$$

*and* $C, H > 0$ *chosen as per Lemma 13 for given* $\delta > 0$*, the resulting stationary policy is stable.*

As discussed earlier, as $\alpha \to 0$, with above choice $\delta = \Theta(\alpha^2)$, the number of samples required per time step scales as $O\left(\frac{1}{\alpha^{2d+4}}\log^{d+1}\frac{1}{\alpha}\right)$. That is, the sample complexity per time step is improved to be polynomial in the Lyapunov drift parameter $\alpha$, rather than super-polynomial as required in Theorem 9. We remark that as the modification replaces states by their closet elements in $\mathcal{S}_\varepsilon$ (and hence, all states in the simulation tree are in $\mathcal{S}_\varepsilon$), the above result on sample complexity provably demonstrates that appropriate data reuse can be beneficial in the algorithmic designs of data efficient learning methods.

### ■ 4.3.3 Discovering Appropriate Policy Parameter

The sample inefficient policy described in Section 4.3.1 or the sample efficient policy described in Section 4.3.2 are stable only if the parameter $\delta$ (or equivalently $\tau$, since $\delta = \tau^2$ in Theorem 9 or 10) is chosen to be small enough. However, what is small enough value for $\delta$ is not clear a priori without knowledge of the system parameters as stated in (4.7) or (4.9). In principle, we can continue reducing $\delta$; however, the sample complexity required per unit time increases with reduction in $\delta$. Therefore, it is essential to ensure that the reduction is eventually terminated. Below we describe such a method, based on a hypothesis test using the boundedness property of stability established in the proofs of Theorems 9 and 10.

**A Statistical Hypothesis Test.** Towards the goal of finding an appropriate value of $\delta$, i.e., small enough but not too small, we describe a statistical hypothesis test — if $\delta$ is small enough, then test passes with high probability. We shall utilize this test to devise an adaptive method that finds the right value of $\delta$ as described below. We state the following structural assumption.

**Condition 4.** *Consider the setting of Condition 2. Let the Lyapunov function $L : \mathcal{S} \to \mathbb{R}_+$, in addition, satisfy*

$$L(s) \geq c_1 \|s\| + c_2, \tag{4.10}$$

*for all $s \in \mathcal{S}$ with constants $c_1, c_2 \in \mathbb{R}$ and $c_1 > 0$.*

Let $\tau(\alpha)$ be as defined in (4.7) and $\delta(\alpha) = \tau(\alpha)^2$. Under the above condition, arguments used in the proofs of Theorem 9 and 10 establish that if $\delta \leq \delta(\alpha)$, then the following exponential tail bound holds:

$$\mathbb{P}\left(\|s_t\| \geq b\right) \leq Ce^{-\eta c_1 b}, \quad \forall\, b > 0, t \geq 1.$$

Note that the probability bound on the right hand side is summable over $t$ when choosing, say, $b = \log^2 t$. This property enables a hypothesis test, via checking $\|s_t\|$, which is used below to devise an adaptive method. We remark that the norm in use is not necessarily $\ell_2$ norm; it can be an arbitrary norm since all norms are equivalent in a finite-dimensional vector space.

**An Adaptive Method for Tuning $\delta$.** Using the statistical hypothesis test, we now describe an algorithm that finds $\delta$ under which the hypothesis test is satisfied with probability exponentially close to 1, and $\delta$ is strictly positive. Initially, set $\delta_0 = 1$. At each time $t$, we decide to whether adjust value of $\delta$ or not by checking whether $\|s_t\| \geq \log^2 t$. If yes, then we set $\delta_{t+1} = \delta_t/2$; else we keep $\delta_{t+1} = \delta_t$.

**Stability Guarantees.** The following theorem provides guarantees for the above adaptive algorithm. The proof is provided in Section 4.6.

**Theorem 11.** *Consider the setup of Theorem 9 (respectively Theorem 10). Let, in addition, Condition 4 hold. Consider the system operating with choice of parameter $\delta = \tau^2$ at any time as per the above described method with policy described in Section 4.3.1 (respectively Section 4.3.2). Then with probability 1, the system operating under such changing choice of $\delta$ is either eventually operating with value $\delta \leq \delta(\alpha)$ and hence stable, or*

$$\limsup_{t \to \infty} \frac{\|s_t\|}{\log^2 t} = O(1); \tag{4.11}$$

*moreover, we have*

$$\liminf_{t\to\infty} \delta_t > 0. \tag{4.12}$$

The theorem guarantees that the system is either stable as the algorithm finds the appropriate policy parameter, or near-stable in the sense that the state grows at most as $\log^2 t$. That is, this adaptive algorithm induces at worst $O(\log^2 t)$ regret since the optimal policy will retain $\|s_t\| = O(1)$, i.e., the algorithm is indeed a low-regret algorithm in that respect.

More precisely, as a corollary of Theorem 11, we have the following result on the sublinear growth rate of $\|s_t\|$. We prove this corollary in Section 4.6.2.

**Corollary 1.** *Consider the setup of Theorem 11. Then with probability 1, we have*

$$\lim_{t\to\infty} \frac{\|s_t\|}{t} = 0. \tag{4.13}$$

**Connecting to Rate Stability.** With the proposed algorithms achieving stability for generic MDPs, let us now re-visit queueing systems to understand the implication of our framework and results. In the context of queueing systems, $s_t$ is often the vector of queue lengths. Corollary 1 suggests that the queue lengths grow "sub-linearly" in $t$, which guarantees *rate stability* for the queueing system.

As a concrete example, consider the multi-queue single-server system introduced in Section 4.2.2. Let $I_{k,t}$ denote the cumulative number of jobs that have arrived at queue $k$ up to time step $t$. We adopt the convention that $I_{k,0} = 0$. The stationary arrival processes $\{I_{k,\cdot}, k \in [K]\}$ satisfy a strong law of large numbers: with probability one,

$$\lim_{t\to\infty} \frac{I_{k,t}}{t} = \lambda_k, \qquad k \in [K]. \tag{4.14}$$

Let $D_{k,t}$ denote the cumulative number of departures from queue $k$ up to time $t$. We also assume that $D_{k,0} = 0$. Note that the system state $s_t$, i.e., the queue-length vector $s_t = (s_{1,t}, s_{2,t}, \ldots, s_{K,t})$, satisfies the following equations of evolution in terms of the total number of arrivals at and departures from each queue:

$$s_{k,t} = s_{k,0} + I_{k,t} - D_{k,t}, \qquad t \geq 0, \forall\, k \in [K]. \tag{4.15}$$

As discussed in Section 4.2.2, the system satisfies Condition 1 and Condition 2 with Lyapunov function $L(s) = \sum_{k\in[K]} \frac{s_k}{\mu_k}$. We further remark that the Lyapunov function $L$ also satisfies Condition 4 with $\|\cdot\|$ being $\ell_1$ norm. Therefore, Theorem 11 and Corollary 1 apply to the example. By Corollary 1, we can further show that the queueing system operating under the adaptive algorithm is *rate stable*, i.e., the "departure rate" of jobs is the same as

the "arrival rate" of jobs [46], as shown below.

Consider the above described queueing system operating under the adaptive method with policy described in Section 4.3.3. Suppose that the initial state $s_0$ is bounded almost surely. As Corollary 1 applies, with probability 1, we have $\lim_{t\to\infty} \frac{\|s_t\|}{t} = 0$. Therefore, with probability 1,

$$\lim_{t\to\infty} \frac{s_{k,t}}{t} = 0, \qquad \forall\, k \in [K]. \tag{4.16}$$

By the system state evolution equation (4.15), for each $k \in [K]$, we have

$$\begin{aligned} \lim_{t\to\infty} \frac{D_{k,t}}{t} &= \lim_{t\to\infty} \frac{s_{k,0} + I_{k,t} - s_{k,t}}{t} \\ &= \lambda_k, \qquad a.s., \end{aligned} \tag{4.17}$$

where the second equality follows from the assumptions on the initial state $s_0$ and the arrival process (4.14), and (4.16).

The above result (4.17) states that the system is rate stable, which is a highly desirable guarantee for queueing systems and can be achieved by our online and adaptive RL policy. We remark that the rate stable result also holds for the single-queue example. Through concrete queueing examples, we demonstrate the merit of our framework of developing stable policies for abstract MDPs, and how it can be naturally adapted to obtain desired properties in specific applications.

### ■ 4.3.4 Discussion

The development of our theory has been devoted to the infinite-horizon discounted reward setting, which is the focus of majority of the RL theory literature. This chapter hence provides a generic framework with a broader application. When specialized to queueing systems, however, there is a small discrepancy: performance criteria for queueing systems such as average delay may be more suitable to be formulated as an average reward problem. In particular, for some queueing systems, the queue lengths may grow unboundedly under the policy that is optimal with respect to the discounted reward [25]. That is, the system is not stable under the discounted optimal policy. This means that while being intuitive, Condition 2 on the discounted optimal policy can be violated. The queueing examples we have discussed in this chapter do not have this issue, as for these systems, the discounted optimal policy is also optimal for the average reward. Addressing this discrepancy in the context of queueing systems would require joint efforts from the RL and queueing communities: (1) from a queueing perspective, it would be desirable to characterize cases/conditions where the discounted optimal policy coincides with the "average" optimal policy; or more

generally, when and how an appropriate reward design would capture the desired properties of queueing systems while the MDP under the optimal policy is positive recurrent— the current framework would apply seamlessly for such scenarios; (2) from a RL viewpoint, we note that the methodology developed in this work would apply for the case of average reward, provided that we have access to a Monte Carlo oracle that produces accurate estimation of $Q$-values under the average reward setting. In other words, the current framework can be extended to develop an online stable policy, by replacing the Monte Carlo oracle with one that targets average reward. Advancement in such an oracle from the RL community would naturally improve the design of stable policy for queueing systems. We believe this work can serve as a starting point for developing principled improvement of reinforcement learning techniques for queueing networks.

## ■ 4.4 Proof of Theorem 9

The remaining sections are devoted to the proofs of our technical results. We start with Theorem 9 in this section. The proof of Theorem 9 is based on three lemmas (Lemmas 14–16). We defer the proofs of these lemmas to subsections 4.4.1–4.4.3.

For any given $\delta > 0$, we know that at each step, with probability $1 - \delta$,

$$\left|Q^*(s_t, a) - \widehat{Q}(s_t, a)\right| \leq \varepsilon \triangleq \frac{2R_{\max}}{1-\gamma}\delta, \quad \forall a \in \mathcal{A}, \tag{4.18}$$

with appropriate choice of parameters $C, H$ as stated in Lemma 12. The stationary policy utilizes Boltzman transformation of $\widehat{Q}$. The following lemma establishes the approximation error between the Boltzman policy and the optimal policy.

**Lemma 14.** *Given state $s \in \mathcal{S}$, let $\widehat{Q}(s, a)$ be such that with probability at least $1 - \delta$,*

$$\left|\widehat{Q}(s, a) - Q^*(s, a)\right| \leq \varepsilon, \quad \forall\, a \in \mathcal{A}.$$

*Consider two Boltzmann policies with temperature $\tau$ :*

$$\widehat{P}(s, a) = \frac{e^{\widehat{Q}(s,a)/\tau}}{\sum_{a'} e^{\widehat{Q}(s,a')/\tau}} \quad \text{and} \quad P^*(s, a) = \frac{e^{Q^*(s,a)/\tau}}{\sum_{a'} e^{Q^*(s,a')/\tau}}, \quad \forall\, a \in \mathcal{A}.$$

*Then, we have that*

1. *With probability at least $1 - \delta$,*

$$\left\|\widehat{P}(s) - P^*(s)\right\|_{\text{TV}} \leq \frac{|\mathcal{A}|^2}{2} \cdot \frac{e^{2\varepsilon/\tau} - 1}{e^{2\varepsilon/\tau} + 1}.$$

*2. With probability at least* $1-\delta$,

$$\left\|\widehat{P}(s)-\pi^*(s)\right\|_{\mathrm{TV}} \le \frac{|\mathcal{A}|^2}{2}\cdot\frac{e^{2\varepsilon/\tau}-1}{e^{2\varepsilon/\tau}+1} + \big(|\mathcal{A}|-1\big)e^{-\frac{\Delta_{\min}(s)}{\tau}}.$$

From Lemma 14, with notation $\Delta_{\min} \le \Delta_{\min}(s)$ for any $s \in \mathcal{S}$, we obtain that for our stochastic policy $\widehat{P}$ at time $t$, with probability $1-\delta$,

$$\left\|\widehat{P}(s_t)-\pi^*(s_t)\right\|_{\mathrm{TV}} \le \kappa \triangleq \frac{|\mathcal{A}|^2}{2}\cdot\frac{e^{2\varepsilon/\tau}-1}{e^{2\varepsilon/\tau}+1} + \big(|\mathcal{A}|-1\big)e^{-\frac{\Delta_{\min}}{\tau}}. \tag{4.19}$$

By Condition 2, we know that the MDP dynamics under the optimal policy respects a Lyapunov function that has bounded increment and drift properties. As per (4.19), the Boltzman policy is a good approximation of the optimal policy at each time step with high probability. The following lemma argues that under such an approximate policy, MDP respects the same Lyapunov function but with a slightly modified drift condition.

**Lemma 15.** *Consider the set up of Theorem 9. Suppose that at each time step* $t$*, a stochastic policy* $\pi(s_t)$ *(i.e.,* $\pi(s_t)$ *is a distribution over* $\mathcal{A}$*) is executed such that for each* $t$*, with probability at least* $1-\delta$,

$$\|\pi(s_t)-\pi^*(s_t)\|_{\mathrm{TV}} \le \kappa.$$

*Then, for every* $s \in \mathcal{S}$ *such that* $L(s) > B$*, we have*

$$\mathbb{E}[L(s_{t+1})-L(s_t)|s_t=s] \le 4\nu\big((1-\delta)\kappa+\delta\big)-\alpha.$$

Now, based on Lemma 15, we note that for every $s \in \mathcal{S}$ such that $L(s) > B$, the following drift inequality holds for our stochastic policy $\widehat{P}$:

$$\mathbb{E}[L(s_{t+1})-L(s_t)|s_t=s] \le 4\nu\big((1-\delta)\kappa+\delta\big)-\alpha. \tag{4.20}$$

We shall argue that under choice of $\delta=\tau^2$ and $\tau$ as per (4.7), the right hand side of (4.20) is less than $-\frac{1}{2}\alpha$. To do so, it is sufficient to argue that $4\nu(\kappa+\delta) \le \alpha/2$. That is, we want to argue

$$4\nu\Big(\underbrace{\frac{|\mathcal{A}|^2}{2}\cdot\Big(1-\frac{2}{e^{4R_{\max}\delta/(\tau(1-\gamma))}+1}\Big)}_{\text{(I)}} + \underbrace{\big(|\mathcal{A}|-1\big)e^{-\frac{\Delta_{\min}}{\tau}}}_{\text{(II)}} + \underbrace{\delta}_{\text{(III)}}\Big) \le \frac{1}{2}\alpha. \tag{4.21}$$

To establish the above claim, it is sufficient to argue that each of (I), (II) and (III) is no more than $\alpha/24\nu$, under the choice of $\tau$ as per (4.7) and $\delta=\tau^2$. To that end, (III) is less than $\alpha/24\nu$ immediately since $\tau \le \sqrt{\frac{\alpha}{24\nu}}$. For (II), similar claim follows due to $\tau \le \frac{\Delta_{\min}}{\ln\left(\frac{24\nu|\mathcal{A}|}{\alpha}\right)}$.

For (I), using facts that $e^x \leq 1+2x,\ x \in (0,1)$ and $1/(1+x) > 1-x$ for $x \in (0,1)$ and $4R_{\max}\delta/(\tau(1-\gamma)) = 4R_{\max}\tau/(1-\gamma) \leq \frac{1}{2}$, we have that (with $\delta = \tau^2$)

$$\begin{aligned}\frac{|\mathcal{A}|^2}{2} \cdot \Big(1 - \frac{2}{e^{4R_{\max}\delta/(\tau(1-\gamma))}+1}\Big) &\leq \frac{|\mathcal{A}|^2}{2} \cdot \Big(1 - \frac{2}{8R_{\max}\tau/(1-\gamma)+2}\Big) \\ &\leq \frac{|\mathcal{A}|^2}{2} \cdot \big(4R_{\max}\tau/(1-\gamma)\big) \leq \frac{\alpha}{24\nu}.\end{aligned}$$

Thus, we conclude that if $L(s) > B$, then

$$\mathbb{E}[L(s_{t+1}) - L(s_t)|s_t = s] \leq -\frac{1}{2}\alpha. \tag{4.22}$$

We recall the following result of [71] that implies positive recurrence property for stochastic system satisfying drift condition as in (4.22).

**Lemma 16.** *Consider a policy $\pi$. Suppose that there exists a Lyapunov function $L$ such that the policy $\pi$ satisfies the bounded increment condition with parameter $\nu > 0$ and the drift condition with parameters $\alpha > 0$ and $B > 0$. Let $T_a \triangleq \min\{t \geq 0 : L(s_t) \leq a\}$. Let $c(\nu) = e^\nu - \nu - 1$, $\eta = \min(1, \alpha/2c(\nu))$ and $\rho = 1 - \eta\alpha/2c(\nu) < 1$. Then it follows that for all $b \geq 0$,*

$$\mathbb{P}(L(s_t) \geq b|s_0 = s) \leq \rho^t e^{\eta(L(s)-b)} + \frac{1-\rho^t}{1-\rho} e^{\nu+\eta(B-b)}, \tag{4.23}$$

$$\mathbb{P}(T_a > k|s_0 = s) \leq e^{\eta(L(s)-a)}\rho^k, \quad \forall\, a \geq B. \tag{4.24}$$

By an immediate application of Lemma 16, with $c = c(\nu) = e^\nu - \nu - 1$, $\eta = \min(1, \alpha/4c(\nu))$ and $\rho = 1 - \eta\alpha/4c(\nu) < 1$, it follows that

$$\mathbb{P}\big(L(s_t) \geq b|s_0 = s\big) \leq \rho^t e^{\eta(L(s)-b)} + \frac{1-\rho^t}{1-\rho} e^{\eta(B-b)+\nu}, \quad \forall\, s \in \mathcal{S}, \tag{4.25}$$

and $\forall\, b \geq B$,

$$\mathbb{P}\big(T_b(t) > k \mid s_t = s\big) \leq e^{\eta(L(s)-b)}\rho^k, \quad \forall\, s \in \mathcal{S}, \tag{4.26}$$

where $T_b(t) \triangleq \min\{k \geq 0 \ :\ L(s_{t+k}) \leq b\}$ is the return time to a set that the Lyapunov function is bounded by $b$ starting from time $t$.

Define level set $\mathcal{C}(\ell) \triangleq \{s \in \mathcal{S} \ :\ L(s) \leq \ell\}$. By Condition 2, $\sup_{s\in\mathcal{C}(\ell)} \|s\| < \infty$ for any finite $\ell$. Now (4.25) implies that for any small $\phi > 0$, we have

$$\limsup_{t\to\infty} \mathbb{P}\big(s_t \notin \mathcal{C}(b+\phi)|s_0 = s\big) \leq \limsup_{t\to\infty} \mathbb{P}\big(L(s_t) \geq b|s_0 = s\big) \leq \frac{1}{1-\rho} e^{\eta(B-b)+\nu}.$$

By letting $b \geq B$ and $b$ be large enough, we can always make the above probability bound as small as possible. That is, for any given $\theta > 0$, there exist a large $b \geq B$ and a small $\phi > 0$, such that

$$\lim_{t\to\infty} \mathbb{P}\big(s_t \notin \mathcal{C}(b+\phi) | s_0 = s\big) \leq \theta, \quad \forall\, s \in \mathcal{S}.$$

In addition, (4.26) implies that

$$\mathbb{P}\big(T_{b+\phi}(t) > k \mid s_t = s\big) \leq e^{\eta\left(L(s)-b-\phi\right)} \rho^k, \quad \forall\, s \in \mathcal{S} \text{ and } \forall\, t \geq 0.$$

Therefore,

$$\mathbb{E}\big[T_{b+\phi}(t) | s_t = s\big] = \sum_{k=0}^{\infty} \mathbb{P}\big(T_{b+\phi}(t) > k | s_t = s\big) \leq e^{\eta(L(s)-b-\phi)} \cdot \frac{1}{1-\rho} < \infty.$$

That is, given the current state $s \in \mathcal{S}$, the return time to the bounded set $\mathcal{C}(b+\phi)$ is uniformly bounded, across all $t$. This establishes the stability of the policy as desired in Theorem 9. □

## ■ 4.4.1 Proof of Lemma 14

We first bound the total variation distance between $\widehat{P}$ and $P^*$. For each $a$, we have

$$\begin{aligned}
\left|\widehat{P}(s,a) - P^*(s,a)\right| &= \left| \frac{e^{\widehat{Q}(s,a)/\tau}}{\sum_{a'} e^{\widehat{Q}(s,a')/\tau}} - \frac{e^{Q^*(s,a)/\tau}}{\sum_{a''} e^{Q^*(s,a'')/\tau}} \right| \\
&= \frac{\left| e^{\widehat{Q}(s,a)/\tau} \sum_b e^{Q^*(s,b)/\tau} - e^{Q^*(s,a)/\tau} \sum_b e^{\widehat{Q}(s,b)/\tau} \right|}{\left(\sum_{a'} e^{\widehat{Q}(s,a')/\tau}\right)\left(\sum_{a''} e^{Q^*(s,a'')/\tau}\right)} \\
&= \sum_b \frac{\left| e^{\widehat{Q}(s,a)/\tau + Q^*(s,b)/\tau} - e^{Q^*(s,a)/\tau + \widehat{Q}(s,b)/\tau} \right|}{\left(\sum_{a'} e^{\widehat{Q}(s,a')/\tau}\right)\left(\sum_{a''} e^{Q^*(s,a'')/\tau}\right)}.
\end{aligned}$$

Consider $b$-th term in the above summation:

$$\begin{aligned}
&\frac{\left| e^{\widehat{Q}(s,a)/\tau + Q^*(s,b)/\tau} - e^{Q^*(s,a)/\tau + \widehat{Q}(s,b)/\tau} \right|}{\left(\sum_{a'} e^{\widehat{Q}(s,a')/\tau}\right)\left(\sum_{a''} e^{Q^*(s,a'')/\tau}\right)} \leq \frac{\left| e^{\widehat{Q}(s,a)/\tau + Q^*(s,b)/\tau} - e^{Q^*(s,a)/\tau + \widehat{Q}(s,b)/\tau} \right|}{e^{\widehat{Q}(s,a)/\tau + Q^*(s,b)/\tau} + e^{Q^*(s,a)/\tau + \widehat{Q}(s,b)/\tau}} \\
&\leq \frac{e^{Q^*(s,a)/\tau + \widehat{Q}(s,b)/\tau} \cdot \left| e^{\left[\widehat{Q}(s,a) + Q^*(s,b) - Q^*(s,a) - \widehat{Q}(s,b)\right]/\tau} - 1 \right|}{e^{Q^*(s,a)/\tau + \widehat{Q}(s,b)/\tau} \cdot \left( e^{\left[\widehat{Q}(s,a) + Q^*(s,b) - Q^*(s,a) - \widehat{Q}(s,b)\right]/\tau} + 1 \right)} \\
&= \frac{\left| e^{\left[\widehat{Q}(s,a) + Q^*(s,b) - Q^*(s,a) - \widehat{Q}(s,b)\right]/\tau} - 1 \right|}{e^{\left[\widehat{Q}(s,a) + Q^*(s,b) - Q^*(s,a) - \widehat{Q}(s,b)\right]/\tau} + 1} = \frac{|e^t - 1|}{e^t + 1},
\end{aligned}$$

where

$$t := \left[\widehat{Q}(s,a) + Q^*(s,b) - Q^*(s,a) - \widehat{Q}(s,b)\right]/\tau.$$

By assumption, we have $|t| \le 2\varepsilon/\tau$. Now, if $t > 0$, then

$$\frac{\left|e^t - 1\right|}{e^t + 1} = \frac{e^t - 1}{e^t + 1} = 1 - \frac{2}{e^t + 1} \le 1 - \frac{2}{e^{2\varepsilon/\tau} + 1} = \frac{e^{2\varepsilon/\tau} - 1}{e^{2\varepsilon/\tau} + 1}.$$

Similarly, if $t < 0$, then

$$\frac{\left|e^t - 1\right|}{e^t + 1} = \frac{-e^t + 1}{e^t + 1} = -1 + \frac{2}{e^t + 1} \le -1 + \frac{2}{e^{-2\varepsilon/\tau} + 1} = \frac{-e^{-2\varepsilon/\tau} + 1}{e^{-2\varepsilon/\tau} + 1} = \frac{-1 + e^{2\varepsilon/\tau}}{e^{2\varepsilon/\tau} + 1}.$$

That is, for any $|t| \le 2\varepsilon/\tau$, we have

$$\frac{\left|e^t - 1\right|}{e^t + 1} \le \frac{e^{2\varepsilon/\tau} - 1}{e^{2\varepsilon/\tau} + 1}.$$

Combining the above results, we obtain that

$$\left|\widehat{P}(s,a) - P^*(s,a)\right| \le \sum_b \frac{e^{2\varepsilon/\tau} - 1}{e^{2\varepsilon/\tau} + 1} \le |\mathcal{A}| \cdot \frac{e^{2\varepsilon/\tau} - 1}{e^{2\varepsilon/\tau} + 1}.$$

Therefore, we have

$$\left\|\widehat{P}(s) - P^*(s)\right\|_{\mathrm{TV}} = \frac{1}{2}\sum_{a\in\mathcal{A}}\left|\widehat{P}(s,a) - P^*(s,a)\right| \le \frac{|\mathcal{A}|^2}{2} \cdot \frac{e^{2\varepsilon/\tau} - 1}{e^{2\varepsilon/\tau} + 1}.$$

Next, we bound $\|P^*(s) - \pi^*(s)\|_{\mathrm{TV}}$. Let $a^* \in \mathcal{A}^*(s)$. Notice that for each $a \in \mathcal{A}^*(s)$, $P^*(s,a) = P^*(s,a^*) \le \frac{1}{|\mathcal{A}^*(s)|}$. For each $a \notin \mathcal{A}^*(s)$, we have

$$P^*(s,a) = \frac{e^{Q^*(s,a)/\tau}}{\sum_{a'} e^{Q^*(s,a')/\tau}} \le \frac{e^{Q^*(s,a)/\tau}}{e^{Q^*(s,a^*)/\tau}} = e^{-\frac{Q^*(s,a^*) - Q^*(s,a)}{\tau}} \le e^{-\Delta_{\min}(s)/\tau}.$$

Note that for each $a \in \mathcal{A}^*(s)$, $\pi^*(a|s) = \frac{1}{|\mathcal{A}^*(s)|}$, and for each $a \notin \mathcal{A}^*(s)$, $\pi^*(a|s) = 0$. It hence follows that

$$\begin{aligned}
\|P^*(s) - \pi^*(s)\|_{\mathrm{TV}} &= \frac{1}{2}\sum_{a\in\mathcal{A}}|P^*(s,a) - \pi^*(a|s)| \\
&= \frac{1}{2}\sum_{a\in\mathcal{A}^*(s)}\big(\pi^*(a|s) - P^*(s,a)\big) + \frac{1}{2}\sum_{a\notin\mathcal{A}^*(s)} P^*(s,a) \\
&= \sum_{a\notin\mathcal{A}^*(s)} P^*(s,a) \le \sum_{a\notin\mathcal{A}^*(s)} e^{-\Delta_{\min}(s)/\tau}
\end{aligned}$$

$$\leq (|\mathcal{A}|-1)e^{-\Delta_{\min}(s)/\tau}.$$

By triangle inequality, we have

$$\begin{aligned}\left\|\widehat{P}(s)-\pi^*(s)\right\|_{\mathrm{TV}} &\leq \left\|\widehat{P}(s)-P^*(s)\right\|_{\mathrm{TV}} + \|P^*(s)-\pi^*(s)\|_{\mathrm{TV}} \\ &\leq \frac{|\mathcal{A}|^2}{2}\cdot\frac{e^{2\varepsilon/\tau}-1}{e^{2\varepsilon/\tau}+1} + (|\mathcal{A}|-1)e^{-\Delta_{\min}(s)/\tau}.\end{aligned}$$

This concludes the proof of Lemma 14. □

## ■ 4.4.2 Proof of Lemma 15

By Condition 2, for each $s_t \in \mathcal{S}$ we have

$$|L(s_{t+1}) - L(s_t)| \leq \nu,$$

where $s_{t+1} \sim \mathcal{P}(\cdot|s,\pi(s_t))$. Let us analyze the drift of $L$ under the stochastic policy $\pi(s_t)$. Fix a state $s \in \mathcal{S}$ such that $L(s) > B$. Then,

$$\begin{aligned}&\mathbb{E}[L(s_{t+1}) - L(s_t)|s_t = s] \\ =&\Big(\sum_{s'\in\mathcal{S}} L(s')\sum_{a\in\mathcal{A}}\pi(a|s)\mathcal{P}(s'|s,a)\Big) - L(s) \\ =&\Big(\sum_{s'\in\mathcal{S}} L(s')\sum_{a\in\mathcal{A}}\big(\pi(a|s)-\pi^*(a|s)\big)\mathcal{P}(s'|s,a) + \sum_{s'\in\mathcal{S}} L(s')\sum_{a\in\mathcal{A}}\pi^*(a|s)\mathcal{P}(s'|s,a)\Big) - L(s) \\ =&\sum_{a\in\mathcal{A}}\big(\pi(a|s)-\pi^*(a|s)\big)\sum_{s'\in\mathcal{S}} L(s')\mathcal{P}(s'|s,a) + \mathbb{E}_{\pi^*}[L(s_{t+1}) - L(s_t)|s_t = s]. \qquad (4.27)\end{aligned}$$

To analyze the first term of (4.27), we define the sets

$$\mathcal{A}_\pi^+ = \{a \in \mathcal{A} \mid \pi(a|s) \geq \pi^*(a|s)\} \quad \text{and} \quad \mathcal{A}_\pi^- = \mathcal{A}\backslash\mathcal{A}_\pi^+.$$

Note that $\sum_{a\in\mathcal{A}}\pi(a|s) = \sum_{a\in\mathcal{A}}\pi^*(a|s) = 1$. Therefore, we have

$$\begin{aligned}&\sum_{a\in\mathcal{A}}\big(\pi(a|s)-\pi^*(a|s)\big)\sum_{s'\in\mathcal{S}} L(s')\mathcal{P}(s'|s,a) \\ =&\sum_{a\in\mathcal{A}}\big(\pi(a|s)-\pi^*(a|s)\big)\sum_{s'\in\mathcal{S}} L(s')\mathcal{P}(s'|s,a) + \sum_{a\in\mathcal{A}}\big(\pi(a|s)-\pi^*(a|s)\big)L(s) \\ =&\sum_{a\in\mathcal{A}_\pi^+}\big(\pi(a|s)-\pi^*(a|s)\big)\Big[\sum_{s'\in\mathcal{S}} L(s')\mathcal{P}(s'|s,a) - L(s)\Big] \qquad (4.28) \\ &+\sum_{a\in\mathcal{A}_\pi^-}\big(\pi(a|s)-\pi^*(a|s)\big)\Big[\sum_{s'\in\mathcal{S}} L(s')\mathcal{P}(s'|s,a) - L(s)\Big]. \qquad (4.29)\end{aligned}$$

For the term (4.28), note that for each $a$ and $s'$ such that $\mathcal{P}(s'|s,a) > 0$, Condition 2 ensures that

$$|L(s') - L(s)| \le \nu.$$

Consequently, we have

$$\begin{aligned}(4.28) &\le \sum_{a\in\mathcal{A}_\pi^+} \big(\pi(a|s) - \pi^*(a|s)\big)\Big[\sum_{s'\in\mathcal{S}} \big(L(s)+\nu\big)\mathcal{P}(s'|s,a) - L(s)\Big] \\ &= \sum_{a\in\mathcal{A}_\pi^+} \big(\pi(a|s) - \pi^*(a|s)\big)\nu \ \le 2\nu\|\pi(s) - \pi^*(s)\|_{\mathrm{TV}}.\end{aligned}$$

Following similar arguments,

$$(4.29) \le 2\nu\|\pi(s) - \pi^*(s)\|_{\mathrm{TV}}.$$

Combining the above two inequalities, we have

$$\sum_{a\in\mathcal{A}} \big(\pi(a|s) - \pi^*(a|s)\big) \sum_{s'\in\mathcal{S}} L(s')\mathcal{P}(s'|s,a) \le 4\nu\|\pi(s) - \pi^*(s)\|_{\mathrm{TV}}. \tag{4.30}$$

Recall that the total variation distance is bounded by 1 and that the stochastic policy $\pi(s)$ satisfies that with probability at least $1-\delta$, $\|\pi(s) - \pi^*(s)\|_{\mathrm{TV}} \le \kappa$. Taking expectation on both sides of (4.30) with respect to the randomness in the stochastic policy $\pi(s)$, we have

$$\mathbb{E}\left[\sum_{a\in\mathcal{A}} \big(\pi(a|s) - \pi^*(a|s)\big) \sum_{s'\in\mathcal{S}} L(s')\mathcal{P}(s'|s,a)\right] \le 4\nu\big((1-\delta)\kappa + \delta\big). \tag{4.31}$$

Substituting the upper bound (4.31) into (4.27) yields

$$\mathbb{E}\big[L(s_{t+1}) - L(s_t)|s_t = s\big] \le 4\nu\big((1-\delta)\kappa+\delta\big) + \mathbb{E}_{\pi^*}\big[L(s_{t+1}) - L(s_t)|s_t = sb\big].$$

Note that by Condition 2, for each $s \in \mathcal{S}$ such that $L(s) > B$, $\mathbb{E}_{\pi^*}[L(s_{t+1}) - L(s_t)|s_t = s] \le -\alpha$. Finally, we conclude that

$$\mathbb{E}\big[L(s_{t+1}) - L(s_t)|s_t = s\big] \le 4\nu\big((1-\delta)\kappa+\delta\big) - \alpha,$$

thereby completing the proof. □

### ■ 4.4.3 Proof of Lemma 16

Throughout the proof, we fix an $\eta \in (0, \min\{1, \alpha/(2c)\})$ and let $\rho = 1 - \eta\alpha/2$. To simplify the notation, we fix the initial state $s_0$, and the probabilities and expectations should be

understood as conditioned on $s_0 = s$ when appropriate. Let $\mathcal{F}_t$ denote the smallest $\sigma$-algebra containing all information pertaining to the MDP up to time $t$, i.e., $\{s_t\}_{t\geq 0}$ is adapted to $\{\mathcal{F}_t\}_{t\geq 0}$.

We start by establishing (4.23). To do so, we will instead prove the following inequality, from which (4.23) can be readily obtained via Markov's inequality:

$$\mathbb{E}[e^{\eta L(s_t)}] \leq \rho^t e^{\eta L(s_0)} + \frac{1-\rho^t}{1-\rho} e^{\nu+\eta B}. \tag{4.32}$$

Note that $\mathbb{E}[e^{\eta L(s_{t+1})}] = \mathbb{E}[\mathbb{E}[e^{\eta L(s_{t+1})}|\mathcal{F}_t]]$. Further,

$$\mathbb{E}[e^{\eta L(s_{t+1})}|\mathcal{F}_t] = \mathbb{E}[e^{\eta L(s_{t+1})}\mathbb{I}\{L(s_t) \leq B\}|\mathcal{F}_t] + \mathbb{E}[e^{\eta L(s_{t+1})}\mathbb{I}\{L(s_t) > B\}|\mathcal{F}_t]. \tag{4.33}$$

We now analyze the two terms on the R.H.S. of (4.33) separately. For the first term,

$$\mathbb{E}[e^{\eta L(s_{t+1})}\mathbb{I}\{L(s_t) \leq B\}|\mathcal{F}_t] = \mathbb{E}[e^{\eta(L(s_{t+1})-L(s_t))}e^{\eta L(s_t)}\mathbb{I}\{L(s_t) \leq B\}|\mathcal{F}_t] \leq e^{\nu+\eta B},$$

where the last inequality follows from the bounded increment condition and the fact that $\eta < 1$. For the second term of (4.33), observe

$$\begin{aligned}\mathbb{E}[e^{\eta L(s_{t+1})}\mathbb{I}\{L(s_t) > B\}|\mathcal{F}_t] &= \mathbb{E}[e^{\eta(L(s_{t+1})-L(s_t))}e^{\eta L(s_t)}\mathbb{I}\{L(s_t) > B\}|\mathcal{F}_t] \\ &\leq \mathbb{E}[e^{\eta(L(s_{t+1})-L(s_t))\mathbb{I}\{L(s_t)>B\}}|\mathcal{F}_t] \cdot e^{\eta L(s_t)}.\end{aligned}$$

We now show that $\mathbb{E}[e^{\eta(L(s_{t+1})-L(s_t))\mathbb{I}\{L(s_t)>B\}}|\mathcal{F}_t] \leq \rho$. Since $|L(s_{t+1}) - L(s_t)| \leq \nu$ with probability 1, $\mathbb{E}[e^{\eta(L(s_{t+1})-L(s_t))\mathbb{I}\{L(s_t)>B\}}|\mathcal{F}_t]$ has an absolutely convergent series expansion. That is,

$$\begin{aligned}\mathbb{E}[e^{\eta(L(s_{t+1})-L(s_t))\mathbb{I}\{L(s_t)>B\}}|\mathcal{F}_t] &= 1 + \eta\mathbb{E}[(L(s_{t+1}) - L(s_t))\mathbb{I}\{L(s_t) > B\}|\mathcal{F}_t] \\ &\quad + \eta^2 \sum_{k=2}^{\infty} \frac{\eta^{k-2}}{k!}\mathbb{E}[\big((L(s_{t+1}) - L(s_t))\mathbb{I}\{L(s_t) > B\}\big)^k|\mathcal{F}_t] \\ &\leq 1 - \eta\alpha + \eta^2 c,\end{aligned}$$

where we have used the fact that

$$\left|\mathbb{E}[\big((L(s_{t+1}) - L(s_t))\mathbb{I}\{L(s_t) > B\}\big)^k|\mathcal{F}_t]\right| \leq \nu^k$$

for all $k \geq 1$, and notation $c = c(\nu) = e^\nu - \nu - 1$. Note that since $\eta \in (0, \min\{1, \alpha/(2c)\})$,

we have $1 - \eta\alpha + \eta^2 c \leq 1 - \eta\alpha/2 = \rho$. To summarize, we obtain that for (4.33),

$$\mathbb{E}[e^{\eta L(s_{t+1})}|\mathcal{F}_t] \leq e^{\nu+\eta B} + \rho e^{\eta L(s_t)}. \tag{4.34}$$

By taking expectation on both sides of (4.34), we establish the following recursive equation:

$$\mathbb{E}[e^{\eta L(s_{t+1})}] \leq e^{\nu+\eta B} + \rho \mathbb{E}[e^{\eta L(s_t)}].$$

Since (4.32) holds trivially for $t = 0$, the above recursive equation implies that the desired inequality (4.32) is valid for all $t \geq 0$.

Next, we establish (4.24) of Lemma 16. Fix an $a \geq B$ and define $M(t) = \frac{e^{\eta L(s_t)}}{\rho^t}$. Recall that in the above proof, we showed that

$$\mathbb{E}[e^{\eta(L(s_{t+1})-L(s_t))\mathbb{I}\{L(s_t)>B\}}|\mathcal{F}_t] \leq \rho.$$

Therefore, $\{M(\min(t, T_a))\}$ is a non-negative supermartingale. This implies that $M(0) \geq \mathbb{E}[M(\min(t, T_a))]$, i.e.,

$$\begin{aligned} e^{\eta L(s_0)} &\geq \mathbb{E}[e^{\eta L(s_{\min(t,T_a)})}/\rho^{\min(t,T_a)}] \\ &\geq \mathbb{E}[e^{\eta L(s_{\min(t,T_a)})}/\rho^{\min(t,T_a)}\mathbb{I}\{T_a > t\}] \\ &\geq \frac{e^{\eta a}}{\rho^t} \cdot \mathbb{P}(T_a > t), \end{aligned}$$

where the last inequality follows from the definition of $T_a$. This completes the proof of (4.24) and Lemma 16. □

## ■ 4.5 Proof of Theorem 10

The proof of Theorem 10 follows identically as that of Theorem 9 by replacing the performance guarantees of Lemma 12 by that of Lemma 13.

### ■ 4.5.1 Proof of Lemma 13

We establish the statement of Lemma 13 inductively. To begin with, we shall count the total number of samples of the MDP utilized in the algorithm. To that end, note that for the $H$ steps of the procedure starting with root note $s$, any state sampled within $H$ steps from it can not be farther than $H\nu'$ in $\|\cdot\|$ distance per Condition 2. By construction, the number of such states contained in $\mathcal{S}_\varepsilon$ for a given $s$, is at most $N(H, \varepsilon) := O((H\nu'/\varepsilon)^d)$. For each of these $N(H, \varepsilon)$ states, for each action $a \in \mathcal{A}$, we need to sample at most $C$ next states. That is, number of samples are at most $C|\mathcal{A}|N(H, \varepsilon)$. Indeed, as part of our procedure, it

is likely that some of these $N(H,\varepsilon)$ states are visited multiple times.

Now, for any $\tilde{s} \in \mathcal{S}_\varepsilon$ amongst these $N(H,\varepsilon)$ states and for any $a \in \mathcal{A}$, by definition,

$$Q^*(\tilde{s},a) = R(\tilde{s},a) + \gamma \mathbb{E}_{\widehat{s}\sim\mathcal{P}(\cdot|\tilde{s},a)}\big[V^*(\widehat{s})\big]. \tag{4.35}$$

Let $\{\tilde{s}_i\}_{i\le C}$ be $C$ sampled next states per MDP at state $\tilde{s}$ under action $a$. Due to the standard application of Chernoff's bound for bounded valued variables, for any $\lambda > 0$,

$$\mathbb{P}\bigg(\Big|\frac{1}{C}\sum_{i=1}^{C} V^*(\tilde{s}_i) - \mathbb{E}_{\widehat{s}\sim\mathcal{P}(\cdot|\tilde{s},a)}\big[V^*(\widehat{s})\big]\Big| > \lambda\bigg) \le 2\exp\Big(-\frac{\lambda^2 C}{2V_{\max}^2}\Big). \tag{4.36}$$

Here we have used the fact that $\|V^*\|_\infty \le V_{\max}$. By choosing $\lambda^2 = \frac{2V_{\max}^2}{C}\log\big(\frac{2|\mathcal{A}|N(H,\varepsilon)}{\delta}\big)$, the event within the left-hand side holds with probability at least $1 - \frac{\delta}{|\mathcal{A}|N(H,\varepsilon)}$. Therefore, by union bound, the event of (4.36) holds for all the $N(H,\varepsilon)$ states and all action $a \in \mathcal{A}$ pairs. Henceforth in the remainder of the proof, we shall assume that this event holds.

Now, for the given query state $s \in \mathcal{S}$, it is at the root of the sampling tree to produce estimate of $Q^*(s,a)$ for all $a \in \mathcal{A}$. As part of the procedure, we sample $C$ next states for $(s,a)$, which were denoted as $S_{\mathsf{next}}(s,a,C) = \{s_1,\ldots,s_C\}$. We map these states to their closest elements in $\mathcal{S}_\varepsilon$, denoted as $S_\varepsilon(s_1),\ldots,S_\varepsilon(s_C)$. Due to Condition 3, we have that

$$\Big|\frac{1}{C}\sum_{i=1}^{C}(V^*(s_i) - V^*(S_\varepsilon(s_i))\Big| \le \zeta\varepsilon\sqrt{d}. \tag{4.37}$$

As per the method, we produce estimate

$$\widehat{Q}^{(0)}(s,a) = R(s,a) + \gamma\frac{1}{C}\sum_{i=1}^{C}\widehat{V}^{(1)}(S_\varepsilon(s_i)). \tag{4.38}$$

Therefore, we have

$$\begin{aligned}
&\Big|\widehat{Q}^{(0)}(s,a) - Q^*(s,a)\Big| \\
&\le \gamma\Big|\frac{1}{C}\sum_{i=1}^{C}\widehat{V}^{(1)}(S_\varepsilon(s_i)) - \mathbb{E}_{\tilde{s}\sim\mathcal{P}(\cdot|s,a)}\big[V^*(\tilde{s})\big]\Big| \\
&\le \gamma\Big|\frac{1}{C}\sum_{i=1}^{C}\Big(\widehat{V}^{(1)}(S_\varepsilon(s_i)) - V^*(S_\varepsilon(s_i)) + V^*(S_\varepsilon(s_i)) - V^*(s_i) + V^*(s_i)\Big) - \mathbb{E}_{\tilde{s}\sim\mathcal{P}(\cdot|s,a)}\big[V^*(\tilde{s})\big]\Big| \\
&\le \gamma(\mathsf{err}^{(1)} + \zeta\varepsilon\sqrt{d} + \lambda).
\end{aligned} \tag{4.39}$$

Here, $\mathsf{err}^{(1)}$ is the maximum of error in value function estimates for states in level 1 in the

sampling tree, and hence $|\widehat{V}^{(1)}(S_\varepsilon(s_i)) - V^*(S_\varepsilon(s_i))| \le \mathsf{err}^{(1)}$ for all $i \le C$; we have used that event of (4.36) holds and also (4.37). Using this argument recursively and the fact that $\widehat{V}^{(h)}(\tilde{s}) = \max_{a\in\mathcal{A}} \widehat{Q}^{(h)}(\tilde{s}, a)$, we have that for all $1 \le h \le H$,

$$\mathsf{err}^{(h-1)} \le \gamma(\mathsf{err}^{(h)} + \zeta\varepsilon\sqrt{d} + \lambda). \tag{4.40}$$

And at the leaf nodes, by definition we have that $\mathsf{err}^{(H)} \le V_{\max}$. Therefore, we conclude that for all $a \in \mathcal{A}$,

$$\left|\widehat{Q}^{(0)}(s,a) - Q^*(s,a)\right| \le \gamma^H V_{\max} + \frac{\gamma(\zeta\varepsilon\sqrt{d} + \lambda)}{1-\gamma}. \tag{4.41}$$

We choose $\gamma^H \le \delta$ or $H = \lceil \log_\gamma(\delta) \rceil$, $\varepsilon = \frac{\delta V_{\max}(1-\gamma)}{2\zeta\gamma\sqrt{d}}$, and $\lambda \le \frac{\delta V_{\max}(1-\gamma)}{2\gamma}$, i.e.

$$\frac{2V_{\max}^2}{C} \log\big(\frac{2|\mathcal{A}|N(H,\varepsilon)}{\delta}\big) \le \frac{\delta^2 V_{\max}^2 (1-\gamma)^2}{4\gamma^2},$$

or

$$C = \Omega\Big(\frac{\gamma^2}{(1-\gamma)^2\delta^2}\Big(\log\frac{2|\mathcal{A}|}{\delta} + d\log H + d\log\frac{1}{\varepsilon}\Big)\Big) \;=\; \Omega\Big(\frac{1}{\delta^2}\log\frac{1}{\delta}\Big),$$

where $\Omega(\cdot)$ hides constant dependent on $\gamma, |\mathcal{A}|, d, V_{\max}$. Thus, the number of samples utilized is $O\big(C(H\nu'/\varepsilon)^d\big) = O\big(\frac{1}{\delta^{2+d}}\log^{d+1}\frac{1}{\delta}\big)$. □

## ■ 4.6 Proof of Results on Adaptive Methods

We devote this section to the proof of results on the adaptive methods, as stated in Section 4.3.3.

### ■ 4.6.1 Proof of Theorem 11

Let $G$ be the stopping time defined as $G \triangleq \inf\{t \ge 0 : \delta_t \le \delta(\alpha)\}$. As per the method, we start with $\delta_0 = 1$. If $\delta(\alpha) > 1$, then $G = 0$. If $\delta(\alpha) < 1$, then either $G < \infty$ or $G = \infty$. We consider these two cases separately.

**Case 1:** $G < \infty$**.** For each $t \ge G$, we have $\delta_t \le \delta(\alpha)$. In this case, the proof of Theorem 9 suggests that the inequality (4.25) holds; that is, for all $b \ge 0$, and all $t \ge G$,

$$\mathbb{P}(L(s_t) \ge b \mid s_G) \le \rho^{t-G} e^{\eta(L(s_G)-b)} + \frac{1-\rho^{t-G}}{1-\rho} e^{\nu+\eta(B-b)},$$

where $\eta = \min(1, \alpha/4c(\nu))$ with $c(\nu) = e^{\nu} - \nu - 1$, and $\rho = 1 - \eta\alpha/4c(\nu) < 1$. By Condition 4, $L(s) \geq c_1 \|s\| + c_2$, and hence,

$$\mathbb{P}(L(s_t) \geq b \mid s_G) \geq \mathbb{P}(c_1 \|s_t\| + c_2 \geq b \mid s_G) = \mathbb{P}\left(\|s_t\| \geq \frac{b - c_2}{c_1} \mid s_G\right).$$

Combining the last two displayed equations and taking $b = c_1 \log^2 t + c_2$, we obtain that

$$\begin{aligned}\mathbb{P}\left(\|s_t\| \geq \log^2 t \mid s_G\right) &\leq \rho^{t-G} e^{\eta(L(s_G) - c_1 \log^2 t - c_2)} + \frac{1-\rho^{t-G}}{1-\rho} e^{\nu + \eta(B - c_1 \log^2 t - c_2)} \\ &\leq \underbrace{\left[e^{\eta(L(s_G) - c_2)} + \frac{1}{1-\rho} e^{\nu + \eta(B - c_2)}\right]}_{C \equiv C(\nu, \alpha, B, L(s_G))} e^{-\eta c_1 \log^2 t}.\end{aligned}$$

That is, for each $t \geq G$, the likelihood of test failing at time $t$ is $Ce^{-\eta c_1 \log^2 t}$. Since $C \sum_{t=G}^{\infty} e^{-\eta c_1 \log^2 t} < \infty$, the Borel-Cantelli lemma ensures that with probability 1, the test fails for finitely many times. Hence, $\liminf_t \delta_t > 0$ as claimed in (4.12). By definition, when $G < \infty$, the choice of $\delta$ is such that $\delta \leq \delta(\alpha)$. Therefore, by Theorem 9 or 10, the system is stable.

**Case 2: $G = \infty$.** In this case, the system never reaches $\delta \leq \delta(\alpha)$ and hence equation (4.12) holds. In this case, we may not be able to guarantee stability; nevertheless, we can ensure near-stability. Now, since we start with $\delta_0 = 1$ and $G = \infty$, we have $\delta(\alpha) < 1$. Since $G = \infty$, the test fails no more than $\lceil \log_2 1/\delta(\alpha) \rceil = O(1)$ times. Therefore, in the limit of $T \to \infty$, the test does not fail. That is, $\limsup_{t\to\infty} \frac{\|s_t\|}{\log^2 t} = O(1)$ as claimed in (4.11). □

## ■ 4.6.2 Proof of Corollary 1

Corollary 1 is an immediate result from the proof of Theorem 11. Using the same notation as in the proof of Theorem 11, we consider two cases:

**Case 1: $G < \infty$.** In this case, using Borel-Cantelli, we have argued that with probability 1, the test fails for finitely many times. By contradiction, this means that with probability 1, $\limsup_{t\to\infty} \frac{\|s_t\|}{\log^2 t} < 1$; otherwise, if $\limsup_{t\to\infty} \frac{\|s_t\|}{\log^2 t} \geq 1$, the test must have failed infinitely many times. Therefore,

$$\limsup_{t\to\infty} \frac{\|s_t\|}{t} \leq \limsup_{t\to\infty} \frac{\|s_t\|}{\log^2 t} \cdot \limsup_{t\to\infty} \frac{\log^2 t}{t} = 0. \tag{4.42}$$

Since $\limsup_{t\to\infty} \frac{\|s_t\|}{t} \geq 0$, (4.42) implies $\limsup_{t\to\infty} \frac{\|s_t\|}{t} = 0$. Moreover, as $\liminf_{t\to\infty} \frac{\|s_t\|}{t} \geq 0$, and $\liminf_{t\to\infty} \frac{\|s_t\|}{t} \leq \limsup_{t\to\infty} \frac{\|s_t\|}{t} = 0$, we have

$$\lim_{t\to\infty} \frac{\|s_t\|}{t} = 0.$$

**Case 2:** $G = \infty$**.** In this case, we have argued in Theorem 11 that $\limsup_{t\to\infty} \frac{\|s_t\|}{\log^2 t} = O(1)$. Apply the same reasoning as in (4.42) and the following argument completes the proof. □

## ■ 4.7 Chapter Summary

In this chapter, we expand the scope of our investigation in this thesis from bounded domains to unbounded cases. In particular, we consider reinforcement learning for systems with unbounded state space, as motivated by classical queueing systems. To tackle the challenges due to unboundedness of the state space, we propose a natural notion of stability to quantify the "goodness" of policy: the system dynamics under the policy should retain the state in a finite, bounded set with high probability.

Our main technical endeavor in this chapter consists of a series of improvements in designing online policies that achieve the desired stability property. After presenting a simple stable policy, we devise a sample efficient algorithm that improves upon it by exploiting the Lipschitz structure during Monte Carlo simulations. The resulting Modified Sparse Sampling Monte Carlo oracle demonstrates a much more favorable data complexity (i.e., polynomial in the parameter of interest). Since the improved oracle can be viewed as a kind of data reuse method, this confirms the efficacy of such a practical technique in Monte Carlo simulations as we conjectured in the chapter summary of Chapter 2, where we studied Monte Carlo Tree Search. To complete our investigation, we further enhance our algorithms with a hypothesis testing procedure that automatically adjusts the unknown parameter to the desired range.

We note that in general, stability in problems with unbounded state space is of significant importance in classical queueing and control theory, yet this aspect has received relatively less attention in existing reinforcement learning literature. In this chapter, we demonstrate the merits of studying stability through more general, abstract properties of MDPs: when specialized to the context of queueing networks, our framework indeed implies the desired rate stability studied in the queueing systems literature. As reinforcement learning becomes increasingly popular in various application domains, we believe that modeling and achieving stability is critical. The framework introduced in this chapter provides some first steps towards this direction.

Chapter 5

# Conclusions and Future Work

In this thesis, we study the data efficiency of reinforcement learning. Our efforts are focused on two directions. First, with recent achievements of applying RL in practice, we are motivated to develop an in-depth *understanding* on the usage of data for empirically successful algorithms, where theory is currently lacking. Popular MCTS-based algorithm that employs MCTS to generate samples for supervised learning, exemplified by AlphaGo Zero, is an ideal candidate. Chapter 2 hence concentrates on analyzing the data complexity of MCTS. We identify the correct polynomial bonus term in order to utilize the tree structure for estimating the values and characterize the finite-sample guarantees of such a procedure. This subsequently allows us to capture the data complexity of a RL algorithm that combines MCTS with nearest neighbor regression. After developing a better understanding, naturally, the second direction of our effort is centered around *designing* efficient simulation-based RL algorithms. We begin with problems with bounded domains in Chapter 3. Through introducing the perspective of low-rank $Q$-function, we demonstrate how such a structure can be naturally exploited with matrix estimation techniques to devise a new data-efficient RL algorithm. The resulting algorithm, which essentially performs sampling and pseudo-exploration via ME, provides an exponential improvement in data complexity. As a byproduct, we also propose a new ME method with $\ell_\infty$ recovery guarantee, which might be of independent interest. Having completed our journey with bounded domains, in Chapter 4, we consider RL with unbounded state space. To this end, we first address the unique conceptual challenges imposed by the unboundedness of the state space, via proposing an appropriate notion of stability. In essence, we desire the system operating under a stable policy to stay in a bounded region with high probability. By exploiting the Lipschitz structure of the value function, we design an efficient stable policy whose data complexity scales polynomially in the parameter of interest. By incorporating a statistical test on the magnitude of the state, the algorithm is further improved to automatically discover the right tuning parameter. Overall, with new technical tools and frameworks, this thesis contributes to the advancement of the analysis and design of data-efficient RL algorithms.

## ■ 5.1 Future Work

In the summary of each chapter, we outline several open questions related to the corresponding results. Collectively, at a higher level, this thesis also brings us to a point with a variety of fruitful directions.

**Settings Beyond a Generative Model.** This thesis assumes the access to a generative model which one can use to freely sample any state-action transition. This has been an important setting in the theoretical reinforcement learning community for developing insights and characterizing data complexities. Yet, having a precise simulator is not always common in practice, except in applications such as video games where we have full information. Therefore, this naturally connects us to the world of model-based RL where learning the transition model is often of interest. From an applied perspective, we observe that once a high-fidelity model is learned, our algorithmic frameworks can then be applied seamlessly. Model-based RL has gained much more attention in the recent years, and advancements in model learning would naturally benefit our methods. In terms of theory, we would be interested in how the error in the learned transition model is translated to our eventual estimation of the RL quantities, i.e., how robust our frameworks are with respect to the model error. Precisely establishing the relationship and capturing the data complexity is of great theoretical value. Further, the discussion above also implies an approach to extend the current results to the offline RL (or batch RL) setting, by first learning a model with the pre-collected data and then applying our existing frameworks. Finally, one may also be interested in extending part of the results to settings where we cannot freely sample any state-action at will; for example, devising new matrix estimation methods to extend the "low-rank" RL algorithm to the online setup. To summarize, extension beyond the ideal generative world is undoubtedly an important future avenue of research.

**Structures Beyond Value Functions.** Designing data-efficient RL algorithm is important for the wider application of RL in practice. Throughout this thesis, we primarily focus on structures within the value functions such as being Lipschitz and low-rank. There are certainly similar lower-dimensional (or low-rank) structures within other RL quantities that could lead to efficient learning with limited data. For example, the preliminary work [4] considers model-based offline RL with heterogeneous agents under severe data scarcity, i.e., only a single historical trajectory per agent is observed. It is posited that the transition dynamics across agents possesses a structural form: it can be represented as a latent function of latent factors associated with agents, states, and actions. That is, the transition dynamics can be well-approximated by a "low-rank" decomposition of separable agent, state, and action latent functions. This low-rank representation suggests a simple, regularized neural network architecture that empirically allows efficient model learning with the limited data considered.

In general, identifying proper structures and establishing their provable efficiency have been an active research area in theoretical RL. This thesis makes contributions to this trend with new insights that could be valuable for investigating other RL quantities of interest.

Appendix A

# Supplementary Materials for Chapter 2

## A.1 Proof of Proposition 1

The recent work [143] establishes a lower bound on the sample complexity for reinforcement learning algorithms on MDPs. We follow a similar argument to establish a lower bound on the sample complexity for MDPs with deterministic transitions. We provide the proof for completeness. The key idea is to connect the problem of estimating the value function to the problem of non-parametric regression, and then leveraging known minimax lower bound for the latter. In particular, we show that a class of non-parametric regression problem can be embedded in a MDP with deterministic transitions, so any algorithm for the latter can be used to solve the former. Prior work on non-parametric regression [164, 154] establishes that a certain number of observations is necessary to achieve a given accuracy using any algorithms, hence leading to a corresponding necessary condition for the sample size of estimating the value function in a MDP. We now provide the details.

**Step 1. Non-parametric Regression.** Consider the following non-parametric regression problem: Let $\mathcal{S} := [0,1]^d$ and assume that we have $T$ data pairs $(x_1, y_1), \ldots, (x_T, y_T)$ such that conditioned on $x_1, \ldots, x_n$, the random variables $y_1, \ldots, y_n$ are independent and satisfy

$$\mathbb{E}\left[y_t | x_t\right] = f(x_t), \qquad x_t \in \mathcal{S} \tag{A.1}$$

where $f : \mathcal{S} \to \mathbb{R}$ is the unknown regression function. Suppose that the conditional distribution of $y_t$ given $x_t = x$ is a Bernoulli distribution with mean $f(x)$. We also assume that $f$ is 1-Lipschitz continuous with respect to the Euclidean norm, i.e.,

$$|f(x) - f(x_0)| \leq ||x - x_0||, \quad \forall x, x_0 \in \mathcal{S}.$$

Let $\mathcal{F}$ be the collection of all 1-Lipschitz continuous function on $\mathcal{S}$, i.e.,

$$\mathcal{F} = \{h | h \text{ is a 1-Lipschitz function on } \mathcal{S}\}.$$

The goal is to estimate $f$ given the observations $(x_1, y_1), \ldots, (x_T, y_T)$ and the prior knowledge that $f \in \mathcal{F}$.

It is easy to verify that the above problem is a special case of the non-parametric regression problem considered in the work by [154] (in particular, Example 2 therein). Let $\hat{f}_T$ denote an arbitrary (measurable) estimator of $f$ based on the training samples $(x_1, y_1), \ldots, (x_T, y_T)$. By Theorem 1 in [154], we have the following result: there exists a $c > 0$ such that

$$\lim_{T\to\infty} \inf_{\hat{f}_T} \sup_{f\in\mathcal{F}} \mathbb{P}\left( \left\| \hat{f}_T - f \right\|_\infty \geq c\left(\frac{\log T}{T}\right)^{\frac{1}{2+d}} \right) = 1, \tag{A.2}$$

where infimum is over all possible estimators $\hat{f}_T$. Translating this result to the non-asymptotic regime, we obtain the following theorem.

**Theorem 12.** *Under the above stated assumptions, for any number $\delta \in (0, 1)$, there exist $c > 0$ and $T_\delta$ such that*

$$\inf_{\hat{f}_T} \sup_{f\in\mathcal{F}} \mathbb{P}\left( \left\| \hat{f}_T - f \right\|_\infty \geq c\left(\frac{\log T}{T}\right)^{\frac{1}{2+d}} \right) \geq \delta, \qquad \text{for all } T \geq T_\delta.$$

**Step 2. MDP with Deterministic Transitions.** Consider a class of discrete-time discounted MDPs $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$, where

$$\begin{aligned}
&\mathcal{S} = [0, 1]^d,\\
&\mathcal{A} \text{ is finite},\\
&\text{for each } (x, a), \text{ there exists a unique } x' \in \mathcal{S} \text{ such that } \mathcal{P}(x'|x, a) = 1,\\
&r(x, a) = r(x) \text{ for all } a,\\
&\gamma = 0.
\end{aligned}$$

In words, the transition is deterministic, the expected reward is independent of the action taken, and only immediate reward matters.

Let $R_t$ be the observed reward at step $t$. We assume that given $x_t$, the random variable $R_t$ is independent of $(x_1, \ldots, x_{t-1})$, and follows a Bernoulli distribution $\text{Bernoulli}\big(r(x_t)\big)$. The expected reward function $r(\cdot)$ is assumed to be 1-Lipschitz and bounded. It is easy to see that for all $x \in \mathcal{S}$, $a \in \mathcal{A}$,

$$V^*(x) = r(x). \tag{A.3}$$

**Step 3. Reduction from Regression to MDP.** Given a non-parametric regression

problem as described in Step 1, we may reduce it to the problem of estimating the value function $V^*$ of the MDP described in Step 2. To do this, we set

$$r(x) = f(x), \qquad \forall x \in \mathcal{S}$$

and

$$R_t = y_t, \qquad t = 1, 2, \ldots, T.$$

In this case, it follows from equations (A.3) that the value function is given by $V^* = f$. Moreover, the expected reward function $r(\cdot)$ is 1-Lipschitz, so the assumptions of the MDP in Step 2 are satisfied. This reduction shows that the MDP problem is at least as hard as the nonparametric regression problem, so a lower bound for the latter is also a lower bound for the former.

Applying Theorem 12 yields the following result: for any number $\delta \in (0, 1)$, there exist some numbers $c > 0$ and $T_\delta > 0$, such that

$$\inf_{\hat{V}_T} \sup_{V^* \in \mathcal{F}} \mathbb{P}\left( \left\| \hat{V}_T - V^* \right\|_\infty \geq c \left( \frac{\log T}{T} \right)^{\frac{1}{2+d}} \right) \geq \delta, \qquad \text{for all } T \geq T_\delta.$$

Consequently, for any reinforcement learning algorithm and any sufficiently small $\varepsilon > 0$, there exists a MDP problem with deterministic transitions such that in order to achieve

$$\mathbb{P}\left( \left\| \hat{V}_T - V^* \right\|_\infty < \varepsilon \right) \geq 1 - \varepsilon,$$

one must have

$$T \geq C' d \left( \frac{1}{\varepsilon} \right)^{2+d} \log \left( \frac{1}{\varepsilon} \right),$$

where $C' > 0$ is a constant. □

## ■ A.2 Numerical Experiments

While the focus of Chapter 2 is to develop a theoretical understanding of MCTS, we provide simple toy examples as supplements to corroborate our results.

To this end, we design a simple class of deterministic MDPs as follows. For each state $s \in \mathcal{S}$ and each action $a \in \mathcal{A}$, we sample uniformly from $\mathcal{S}$ a state and fix it to be the corresponding next state $s'$. The reward $\mathcal{R}(s, a)$ is a uniformly distributed random variable taking values in $[0, R_{\max}(s, a)]$, where the bound $R_{\max}(s, a)$ is uniformly sampled from the interval $[-3, 3]$ beforehand and is then fixed. We let $|\mathcal{S}| = 20$, $|\mathcal{A}| = 5$ and $\gamma = 0.8$. We then

sample a deterministic MDP from the above class and query a state via the MCTS algorithm with different depth $H$. For selecting an action at each depth, we use the polynomial bonus term (2.4) as emphasized throughout Chapter 2 with $\eta = 1/2$. That is, we choose the action with the highest upper confidence bound in the form of "mean reward $+ C \cdot t_s^{1/4}/t_a^{1/2}$". Here, $t_s$ is the number of times that particular node at depth $h$ has been visited; $t_a$ is the number of times the action $a$ is chosen for that node; $C$ is a constant for controlling exploration and exploitation. For simplicity, we choose the same $C$ for each depth as this is common in practice. The value of the leaf nodes is set to 0. Note that per our theoretical results (Theorem 1 and Section 2.7.5), the output of the MCTS algorithm, in expectation, converges to the value estimate after running $H$ steps of value function iteration starting with $\hat{V} \equiv 0$ for all states. To validate this consistency result, we perform 25 independent queries of MCTS with a selected root state and plot the resulting mean and standard deviation. The value estimate after $H$ steps of value function iteration is used as the "true value" to benchmark the experiments. Figure A.1 shows the results for two tree depth: $H = 7$ (left) and $H = 10$ (right). As expected, the output of MCTS converges to the desired true value. The constant $C$ captures the extent of the exploration-exploitation tradeoff. With smaller $C$, the simulation could be under-explored and the error bars are wider due to occasionally inaccurate estimates for some runs. A larger $C$ implies more exploration; consequently it requires more simulation steps to converge. We note that $C = 1$ seems to be a good choice in this example.

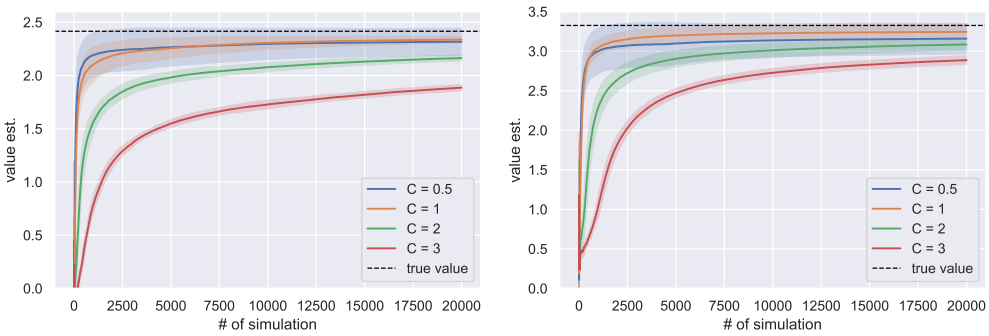



**Figure A.1.** Simulation for a deterministic MDP with tree depth $H = 7$ on the left and $H = 10$ on the right. Each plot is a summary of 25 MCTS experiments showing the mean and standard deviation.

Since our results can be extended to the stochastic environments, we also experiment with stochastic MDPs. The class of stochastic MDPs is constructed in the same manner as before except that for each state $s \in \mathcal{S}$ and each action $a \in \mathcal{A}$: (1) we sample $L$ states uniformly from $\mathcal{S}$ and fix them to be the potential next states; (2) the transition kernel $\mathcal{P}(\cdot|s,a)$ is then sampled from a Dirichlet distribution with $L$ categories. We let $|\mathcal{S}| = 100$, $|\mathcal{A}| = 3$, $L = 3$ and $\gamma = 0.8$. A stochastic MDP is then sampled from the class

and we again perform 25 independent MCTS queries with different depth $H$. Figure A.2 summarizes the corresponding results. A large number of simulation steps is used to account for the additional stochasticity from the transition. Again, these experiments corroborate our theoretical findings, with the mean of the outputs converging to the true value.

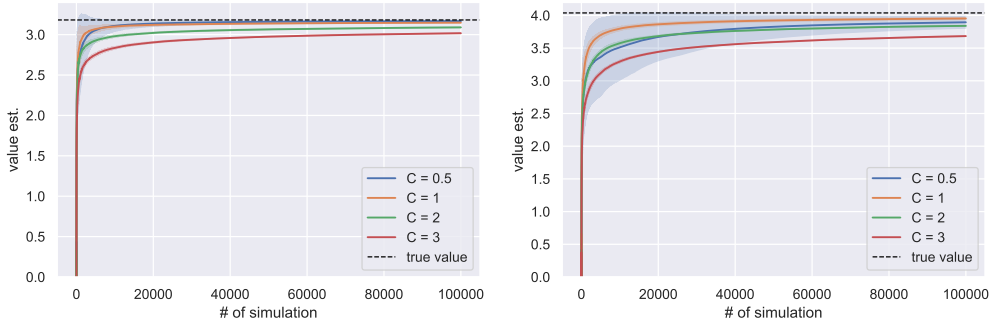



**Figure A.2.** Simulation for a stochastic MDP with tree depth $H = 5$ on the left and $H = 8$ on the right. Each plot is a summary of 25 MCTS experiments showing the mean and standard deviation.

Appendix B

# Supplementary Materials for Chapter 3

## B.1 Proof of Theorem 4

The proof of Theorem 4 follows from the classical results in functional analysis. We present it for completeness. Interested reader may find lecture notes [24] and a classical textbook on the topic [43] as excellent references. In this section, we present and prove a more general version of Theorem 4 that is applicable to any compact metric spaces equipped with finite measures.

Let $\mathcal{S}$ and $\mathcal{A}$ be compact metric spaces, equipped with finite measures $\mu$, $\nu$, respectively. We consider the space of square integrable functions

$$L^2(\mathcal{S},\mu) = \left\{ f : \mathcal{S} \to \mathbb{R} \text{ such that } \|f\|_{L^2(\mathcal{S},\mu)} \equiv \Big( \int_{\mathcal{S}} |f(s)|^2 d\mu(s) \Big)^{\frac{1}{2}} < \infty \right\},$$

and $L^2(\mathcal{A},\nu)$ defined similarly. $L^2(\mathcal{S},\mu)$ and $L^2(\mathcal{A},\nu)$ are known to be Hilbert spaces and in particular, they are separable because $\mathcal{S}$ and $\mathcal{A}$ are compact metric spaces. Therefore, they have countable bases.

Recall that given any vector space $V$ over $\mathbb{R}$, its dual space $V^*$ is defined as the set of all linear maps $\phi : V \to \mathbb{R}$. It is known that the dual of $L^2(\mathcal{S},\mu)$ is isometrically isomorphic to $L^2(\mathcal{S},\mu)$, e.g., by the isomorphism $f \mapsto f^*$ where $f^*(f') = \langle f', f\rangle = \int_{\mathcal{S}} f(s)f'(s)d\mu(s)$ (Appendix B, [43]).

Given two Hilbert spaces, $\mathcal{H}_1, \mathcal{H}_2$, we let $\mathcal{H}_1 \otimes \mathcal{H}_2$ denote the tensor product of the two Hilbert spaces. The inner product in $\mathcal{H}_1 \otimes \mathcal{H}_2$ is defined on the basis elements so that $\langle \phi_1 \otimes \phi_2, \psi_1 \otimes \psi_2\rangle_{\mathcal{H}_1\otimes\mathcal{H}_2} = \langle \phi_1, \psi_1\rangle_{\mathcal{H}_1}\langle \phi_2, \psi_2\rangle_{\mathcal{H}_2}$ for all $\phi_1, \psi_1 \in \mathcal{H}_1$ and $\phi_2, \psi_2 \in \mathcal{H}_2$. Also, for every element $\phi_1 \otimes \phi_2 \in \mathcal{H}_1 \otimes \mathcal{H}_2$, one can associate the rank-1 operator from $\mathcal{H}_1^* \to \mathcal{H}_2$ that maps a given $x^* \in \mathcal{H}_1^*$ to $x^*(\phi_1)\phi_2$.

Our main theorem in this section is the following spectral theorem (singular value theorem) for $Q^*$. It is indeed a classical result from operator theory on Hilbert spaces. However, most results in existing literature cover the theory for self-adjoint operators and symmetric kernels. Although it is already implied by the classical results in a similar manner as eigen-

value decomposition extends to singular value decomposition, here we state our theorem and its proof for readers' convenience and future references.

**Theorem 13.** *Let $(\mathcal{S}, d_{\mathcal{S}}, \mu)$ and $(\mathcal{A}, d_{\mathcal{A}}, \nu)$ be compact metric spaces equipped with finite measures. Let $Q^* \in L^2(\mathcal{S} \times \mathcal{A}, \mu \times \nu)$. If $Q^*$ is $\zeta$-Lipschitz with respect to the product metric, then there exist a nonincreasing sequence $(\sigma_i \geq \mathbb{R}_+ : i \in \mathbb{N})$ with $\sum_{i=1}^{\infty} \sigma_i^2 < \infty$ and orthonormal bases $\{f_i \in L^2(\mathcal{S}, \mu) : i \in \mathbb{N}\}$ and $\{g_i \in L^2(\mathcal{A}, \nu) : i \in \mathbb{N}\}$ such that*

$$Q^* = \sum_{i=1}^{\infty} \sigma_i f_i \otimes g_i. \tag{B.1}$$

*Subsequently, for any $\delta > 0$, there exists $r^*(\delta) \in \mathbb{N}$ such that $\left\| \sum_{i=1}^{r} \sigma_i f_i \otimes g_i - Q^* \right\|^2_{L^2(\mathcal{S}\times\mathcal{A},\mu\times\nu)} \leq \delta$) for all $r \geq r^*(\delta)$.*

Note that we obtain the equality (B.1) in the $L^2$ sense. However, since $Q^*$ is assumed Lipschitz continuous on a compact domain, this actually gives us a pointwise equality, i.e., $Q^*(s,a) = \sum_{i=1}^{\infty} \sigma_i f_i(s) g_i(a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.

*Proof.* We define an integral kernel operator $K = K_{Q^*} : L^2(\mathcal{S}, \mu) \to L^2(\mathcal{A}, \nu)$ induced by the kernel $Q^* \in L^2(\mathcal{S} \times \mathcal{A}, \mu \times \nu)$ so that

$$Kf(\cdot) = \int_{\mathcal{S}} Q^*(s, \cdot) f(s) d\mu(s).$$

Observe that $Q^*$ is a continuous function defined on a compact domain and hence bounded, viz., there exists $V_{\max} < \infty$ such that $|Q^*(s,a)| \leq V_{\max}$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.

We present our proof in four parts. First, we verify that $K$ is a compact operator from $L^2(\mathcal{S}, \mu)$ to $L^2(\mathcal{A}, \nu)$. Next, we argue $K$ admits a generalized singular value decomposition with square summable singular values, based on the spectral theory of compact operators. Then we transfer the results for $K \in L^2(\mathcal{S}, \mu)^* \otimes L^2(\mathcal{A}, \nu)$ to argue the spectral decomposition of $Q^* \in L^2(\mathcal{S}, \mu) \otimes L^2(\mathcal{A}, \nu)$. Lastly, we conclude the proof by discussing rank-$r$ approximation of $Q^*$.

1. $K$ is a compact operator from $L^2(\mathcal{S}, \mu)$ to $L^2(\mathcal{A}, \nu)$.

   First, we argue that $K$ is a bounded linear operator with $\|K\| \leq V_{\max}^2 \mu(\mathcal{S}) \nu(\mathcal{A})$. Recall that $Q^* : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is Lipschitz continuous on a compact domain, hence, bounded, i.e., there exists $V_{\max} < \infty$ such that $|Q^*(s,a)| \leq V_{\max}$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. For any $f \in L^2(\mathcal{S}, \mu)$,

   $$\|Kf\|^2_{L^2(\mathcal{A},\nu)} = \int_{\mathcal{A}} Kf(a)^2 d\nu(a)$$

$$
\begin{aligned}
&= \int_{\mathcal{A}} \left( \int_{\mathcal{S}} Q^*(s,a) f(s) d\mu(s) \right)^2 d\nu(a) \\
&\leq \int_{\mathcal{A}} \|Q^*(\cdot,a)\|^2_{L^2(\mathcal{S},\mu)} \|f\|^2_{L^2(\mathcal{S},\mu)} d\nu(a) \quad \because \text{Cauchy-Schwarz} \\
&\leq V^2_{\max} \mu(\mathcal{S}) \nu(\mathcal{A}) \|f\|^2_{L^2(\mathcal{S},\mu)}. \quad \because \|Q^*(\cdot,a)\|^2_{L^2(\mathcal{S})} \leq V^2_{\max} \mu(\mathcal{S})
\end{aligned}
$$

Next, we show that $K : L^2(\mathcal{S},\mu) \to L^2(\mathcal{A},\nu)$ is indeed a compact operator. It suffices to show that for any bounded sequence $(f_n)_{n\geq 1}$ in $L^2(\mathcal{S},\mu)$, the sequence $(Kf_n)_{n\geq 1}$ contains a convergent subsequence. For this, we use (generalized) Arzelà-Ascoli theorem, which states that if $(Kf_n)_{n\geq 1}$ is uniformly bounded and uniformly equicontinuous, then it contains a convergent subsequence. To that end, first note that $\|Kf_n\| \leq \|K\|\|f_n\|$ and therefore, if $\|f_n\| \leq B$ for all $n \geq 1$, then $\|Kf_n\| \leq \|K\|B$ for all $n \geq 1$. That is, the sequence $(Kf_n)_{n\geq 1}$ is uniformly bounded. Next, we can also verify that $(Kf_n)_{n\geq 1}$ is equicontinuous because for all $n \geq 1$,

$$
\begin{aligned}
\left|Kf_n(a_1) - Kf_n(a_2)\right| &\leq \left| \int_{\mathcal{S}} \left\{Q^*(s,a_1) - Q^*(s,a_2)\right\} f_n(s) d\mu(s) \right| \\
&\leq \|Q^*(s,a_1) - Q^*(s,a_2)\|_{L^2(\mathcal{S})} \|f_n\|_{L^2(\mathcal{S},\mu)} \\
&\leq \zeta \mu(\mathcal{S})^{\frac{1}{2}} d_{\mathcal{A}}(a_1,a_2) \|f_n\|_{L^2(\mathcal{S},\mu)} \\
&\leq B\zeta \mu(\mathcal{S})^{\frac{1}{2}} d_{\mathcal{A}}(a_1,a_2).
\end{aligned}
$$

In the second to last inequality, we used the fact that $Q^*$ is $\zeta$-Lipschitz to show

$$
\begin{aligned}
\|Q^*(s,a_1) - Q^*(s,a_2)\|_{L^2(\mathcal{S},\mu)} \|f_n\|_{L^2(\mathcal{S},\mu)} &= \left( \int_{\mathcal{S}} \left(Q^*(s,a_1) - Q^*(s,a_2)\right)^2 d\mu(S) \right)^{\frac{1}{2}} \\
&\leq \left( \int_{\mathcal{S}} \zeta^2 d_{\mathcal{A}}(a_1,a_2)^2 d\mu(S) \right)^{\frac{1}{2}} \\
&= \zeta \mu(\mathcal{S})^{\frac{1}{2}} d_{\mathcal{A}}(a_1,a_2).
\end{aligned}
$$

2. Spectral decomposition of $K$.

- First of all, we show that there exist orthonormal bases $\{f_i \in L^2(\mathcal{S},\mu) : i \in \mathbb{N}\}$, $\{g_i \in L^2(\mathcal{A},\nu) : i \in \mathbb{N}\}$ and singular values $\{\sigma_i \geq 0 : i \in \mathbb{N}\}$ such that

$$
K = \sum_{i=1}^{\infty} \sigma_i f_i^* \otimes g_i. \tag{B.2}
$$

To see this, we consider the adjoint operator of $K$, namely, $K^* : L^2(\mathcal{A},\nu) \to L^2(\mathcal{S},\mu)$. Since $K : L^2(\mathcal{S},\mu) \to L^2(\mathcal{A},\nu)$ is compact, $K^*$ is also compact. Note

that $K^*K$ is compact and self-adjoint. By the spectral theorem for compact self-adjoint operators, there exist $\{\tau_i \in \mathbb{R} : i \in \mathbb{N}\}$ and an orthonormal basis $\{f_i \in L^2(\mathcal{S}, \mu) : i \in \mathbb{N}\}$ such that $K^*Kf_i = \tau_i f_i$ for all $i \in \mathbb{N}$. We can observe that $\tau_i \geq 0$ for all $i$ because $\tau_i = \tau_i \langle f_i, f_i \rangle = \langle K^*Kf_i, f_i \rangle = \|Kf_i\|^2_{L^2(\mathcal{S},\mu)} \geq 0$. We let $I := \{i \in \mathbb{N} : \tau_i > 0\}$.

Next, we observe that $\ker(K^*K) = \ker(K)$. Showing $\ker(K^*K) \supseteq \ker(K)$ is trivial. To show the other direction, let's suppose that $f \in \ker(K^*K)$. Then $\|Kf\|^2_{L^2(\mathcal{A},\nu)} = \langle Kf, Kf, \rangle = \langle K^*Kf, f \rangle = 0$, which requires $Kf = 0$ and thus $f \in \ker(K)$.

For $i \in I$, we let $g_i = \frac{1}{\sqrt{\tau_i}} Kf_i$. Then $\langle g_i, g_j \rangle = \frac{1}{\sqrt{\tau_i \tau_j}} \langle Kf_i, Kf_j \rangle = \frac{1}{\sqrt{\tau_i \tau_j}} \langle K^*Kf_i, f_j \rangle = \delta_{ij}$, and hence, $\{g_i : i \in I\}$ consists of orthonormal vectors. We can augment $\{g_i : i \in I\}$ by adding appropriate vectors to make $\{g_i : i \in \mathbb{N}\}$ an orthonormal basis of $L^2(\mathcal{A}, \nu)$.

Every vector $\phi \in L^2(\mathcal{S}, \mu)$ can be expanded as $\phi = \sum_{i=1}^\infty \langle \phi, f_i \rangle f_i$. Then we see that $K\phi = \sum_{i=1}^\infty \langle \phi, f_i \rangle Kf_i = \sum_{i=1}^\infty \sqrt{\tau_i} \langle \phi, f_i \rangle g_i$. By letting $\sigma_i = \sqrt{\tau_i}$, we obtain (B.2).

- In addition, we show that $\sum_{i=1}^\infty \sigma_i^2 = \|Q^*\|^2_{L^2(\mathcal{S}\times\mathcal{A},\mu\times\nu)} < \infty$. The Hilbert-Schmidt norm of operator $K$ is defined as $\|K\|_{HS} = \mathrm{Tr}(K^*K) = \sum_{i=1}^\infty \|Kf_i\|^2_{L^2(\mathcal{A},\nu)} < \infty$. Note that $\|K\|_{HS} = \sum_{i=1}^\infty \sigma_i^2$.

  First, we observe that for each $i \in \mathbb{N}$,

$$\begin{aligned}\langle Kf_i, Kf_i \rangle_{L^2(\mathcal{A},\nu)} &= \int_{\mathcal{A}} \left( \int_{\mathcal{S}} Q^*(s,a) f_i(s) d\mu(s) \right)^2 d\nu(a) \\ &= \int_{\mathcal{A}} \left\langle Q^*(\cdot, a), f_i \right\rangle^2_{L^2(\mathcal{S},\mu)} d\nu(a).\end{aligned}$$

  We define a function $G(a) := \left\langle Q^*(\cdot, a), f_i \right\rangle^2_{L^2(\mathcal{S},\mu)}$. Recall that $Q^* \in L^2(\mathcal{S}\times\mathcal{A}, \mu\times\nu)$ and observe that $G$ is a nonnegative measurable function. Then we can use Tonelli's theorem to see that

$$\begin{aligned}\mathrm{Tr}(K^*K) &= \sum_{i=1}^\infty \langle Kf_i, Kf_i \rangle_{L^2(\mathcal{A},\nu)} = \sum_{i=1}^\infty \int_{\mathcal{A}} \left\langle Q^*(\cdot, a), f_i \right\rangle^2_{L^2(\mathcal{S},\mu)} d\nu(a) \\ &= \int_{\mathcal{A}} \sum_{i=1}^\infty \left\langle Q^*(\cdot, a), f_i \right\rangle^2_{L^2(\mathcal{S},\mu)} d\nu(a) \qquad \because \text{Tonelli's theorem} \\ &= \int_{\mathcal{A}} \|Q^*(\cdot, a)\|^2_{L^2(\mathcal{S},\mu)} d\nu(a). \qquad \because \text{the orthonormality of } \{f_i\}\end{aligned}$$

  We have $\int_{\mathcal{A}} \|Q^*(\cdot, a)\|^2_{L^2(\mathcal{S},\mu)} d\nu(a) = \int_{\mathcal{A}} \left( \int_{\mathcal{S}} |Q^*(s,a)|^2 d\mu(s) \right) d\nu(a) = \|Q^*\|^2_{L^2(\mathcal{S}\times\mathcal{A},\mu\times\nu)}$ by Fubini's theorem and therefore, $\sum_{i=1}^\infty \sigma_i^2 = \|Q^*\|^2_{L^2(\mathcal{S}\times\mathcal{A},\mu\times\nu)}$.

3. Spectral decomposition of $Q^*$.

Now we show that $Q^* = \sum_{i=1}^{\infty} \sigma_i f_i \otimes g_i$ for the same singular values $\{\sigma_i \geq 0 : i \in \mathbb{N}\}$ and orthonormal bases $\{f_i \in L^2(\mathcal{S}, \mu) : i \in \mathbb{N}\}$, $\{g_i \in L^2(\mathcal{A}, \nu) : i \in \mathbb{N}\}$ as in (B.2).

For that purpose, we assume that

$$Q^* = \sum_{i=1}^{\infty} \sigma_i f_i \otimes g_i + \varepsilon \tag{B.3}$$

for some $\varepsilon \in L^2(\mathcal{S} \times \mathcal{A}, \mu \times \nu)$. For all $\phi \in L^2(\mathcal{S}, \mu)$ and $\psi \in L^2(\mathcal{A}, \nu)$, we have

$$\begin{aligned}
\langle \psi, K\phi \rangle_{L^2(\mathcal{A},\nu)} &= \int_{\mathcal{A}} \psi(a) \bigg( \int_{\mathcal{S}} Q^*(s,a)\phi(s) d\mu(s) \bigg) d\nu(a) \\
&= \int_{\mathcal{A}} \psi(a) \bigg( \int_{\mathcal{S}} \Big( \sum_{i=1}^{\infty} \sigma_i f_i(s) g_i(a) + \varepsilon(s,a) \Big) \phi(s) d\mu(s) \bigg) d\nu(a) \\
&= \int_{\mathcal{A}} \psi(a) \Big\langle \sum_{i=1}^{\infty} \sigma_i f_i, \ \phi \Big\rangle_{L^2(\mathcal{S},\mu)} g_i(a) d\nu(a) + \int_{\mathcal{A}} \psi(a) \bigg( \int_{\mathcal{S}} \varepsilon(s,a)\phi(s) d\mu(s) \bigg) d\nu(a).
\end{aligned}$$

When $\phi = f_i$ and $\psi = g_j$, we have $\langle g_j, K f_i \rangle_{L^2(\mathcal{A},\nu)} = \sigma_i \delta_{ij}$. By Fubini's theorem,

$$\begin{aligned}
\sigma_i \delta_{ij} &= \sigma_i \langle g_j, g_i \rangle + \int_{\mathcal{S}\times\mathcal{A}} \varepsilon(s,a) f_i(s) g_j(a) d(\mu \times \nu)(s \times a) \\
&= \sigma_i \delta_{ij} + \langle \varepsilon, f_i \otimes g_j \rangle_{L^2(\mathcal{S}\times\mathcal{A}, \mu\times\nu)}.
\end{aligned} \tag{B.4}$$

In order to satisfy (B.4), we must have $\langle \varepsilon, f_i \otimes g_j \rangle_{L^2(\mathcal{S}\times\mathcal{A}, \mu\times\nu)} = 0$ for all $(i,j) \in \mathbb{N}^2$.

It is known that $L^2(\mathcal{S} \times \mathcal{A}, \mu \times \nu)$ is isomorphic to $L^2(\mathcal{S}, \mu) \otimes L^2(\mathcal{A}, \nu)$ and $\{f_i \otimes g_j : (i,j) \in \mathbb{N}^2\}$ constitutes an orthonormal basis of $L^2(\mathcal{S}, \mu) \otimes L^2(\mathcal{A}, \nu)$. Therefore, $\varepsilon = 0$ and $Q^* = \sum_{i=1}^{\infty} \sigma_i f_i \otimes g_i$.

4. Best rank-$r$ approximation of $Q^*$.

   Without loss of generality, we may assume $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$, i.e., the singular values are sorted in descending order. For any finite $r \in \mathbb{N}$, let $Q_r^* = \sum_{i=1}^{r} \sigma_i f_i \otimes g_i$.

   Then,

$$\begin{aligned}
\big\| Q^* - Q_r^* \big\|^2_{L^2(\mathcal{S}\times\mathcal{A}, \mu\times\nu)} &= \Big\| \sum_{i=r+1}^{\infty} \sigma_i f_i \otimes g_i \Big\|^2_{L^2(\mathcal{S}\times\mathcal{A}, \mu\times\nu)} \\
&= \sum_{i,j=r+1}^{\infty} \sigma_i \sigma_j \big\langle f_i \otimes g_i, f_j \otimes g_j \big\rangle_{L^2(\mathcal{S}\times\mathcal{A}, \mu\times\nu)} \\
&= \sum_{i=r+1}^{\infty} \sigma_i^2
\end{aligned}$$

where we have used the orthonormality of $\{f_i\}$ and $\{g_i\}$.

We conclude the proof with two final remarks:

- Among all rank-$r$ functions of the form $\sum_{i=1}^{r} \lambda_i \phi_i \otimes \psi_i$ for some $\phi_i \in L^2(\mathcal{S}, \mu)$, $\psi_i \in L^2(\mathcal{A}, \nu)$, $Q_r^*$ is the "best" rank-$r$ approximation of $Q^*$ in the $L^2(\mathcal{S} \times \mathcal{A}, \mu \times \nu)$ sense.
- Since $\sum_{i=1}^{\infty} \sigma_i^2 < \infty$, for any $\delta > 0$, there exists $r = r(\delta)$ so that $\sum_{i=r+1}^{\infty} \sigma_i^2 < \delta$. That is, we can approximate $Q_r^*$ arbitrarily well with a sufficiently large, yet still finite, rank $r$.

This completes the proof of Theorem 13. □

## ■ B.2 Corollaries of Theorem 7

Recall that our algorithm do not demand any special properties of $\mathcal{S}, \mathcal{A}$ except the existence of $\beta^{(t)}$-net, which is the case whenever $\mathcal{S}, \mathcal{A}$ are compact. Also, our analysis is general in the sense that it only requires $\mathcal{S}, \mathcal{A}$ to be compact with finite measures, and $Q^*$ to be $\zeta$-Lipschitz. Therefore, it is not hard to see that our algorithm and analysis are applicable to the case where state or action space is finite, or both. We summarize results below as corollaries of Theorem 7 without proofs.

Before presenting the results, we recall the following quantity defined in Proposition 3:

$$c(r; \mathcal{S}^\sharp, \mathcal{A}^\sharp) = \left( 6\sqrt{2} \Big( \frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp))} \Big) + 2(1+\sqrt{5}) \Big( \frac{\sqrt{|\mathcal{S}^\sharp||\mathcal{A}^\sharp|}}{\sigma_r(Q^*(\mathcal{S}^\sharp, \mathcal{A}^\sharp))} \Big)^2 \right) V_{\max},$$

that appears in Proposition 3 as the multiplier on the right-hand side of the inequality. This quantity determines the range of $\gamma$ and the convergence rate.

**Continuous $\mathcal{S} \subset \mathbb{R}^d$ and Finite $\mathcal{A}$.** In this case, the algorithm only needs to discretize the state space at each iteration. In other words, $\mathcal{A}^{(t)} = \mathcal{A}$, for all $t = 1, \ldots, T$ and $\Omega^{(t)} = \{(s, a) \in \bar{\mathcal{S}}^{(t)} \times \mathcal{A} : s \in \mathcal{S}^\sharp \text{ or } a \in \mathcal{A}^\sharp\}$. Finally, the generalization step only needs to interpolate over the state space $\mathcal{S}$. We have the following guarantees as an immediate corollary of Theorem 7:

**Corollary 2.** *Consider the rank-$r$ setting with continuous $\mathcal{S}$ and finite $\mathcal{A}$. Suppose that we run the RL algorithm (cf. Section 3.3) with the Matrix Estimation method described in Section 3.5.2. If $\gamma \leq \frac{1}{2c(r; \mathcal{S}^\sharp, \mathcal{A}^\sharp)}$, then the following statements hold.*

1. *For any $\delta > 0$, we have*

$$\sup_{(s,a) \in \mathcal{S} \times \mathcal{A}} \left| Q^{(t)}(s, a) - Q^*(s, a) \right| \leq \big( 2c(r; \mathcal{S}^\sharp, \mathcal{A}^\sharp)\gamma \big)^t V_{\max}, \quad \textit{for all } t = 1, \ldots, T$$

*with probability at least* $1-\delta$ *by choosing algorithmic parameters* $\beta^{(t)}, N^{(t)}$ *appropriately.*

2. *Further, given* $\varepsilon > 0$*, it suffices to set* $T = \Theta(\log \frac{1}{\varepsilon})$ *and use* $\tilde{O}(\frac{1}{\varepsilon^{d+2}} \cdot \log \frac{1}{\delta})$ *number of samples to achieve*

$$\mathbb{P}\left( \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(T)}(s,a) - Q^*(s,a)\right| \le \varepsilon \right) \ge 1-\delta.$$

**Finite $\mathcal{S}$ and Finite $\mathcal{A}$.** Since the spaces are discrete, we have an optimal $Q^*$ being a $|\mathcal{S}| \times |\mathcal{A}|$ matrix. For this special case, the algorithm simply skips the discretization (i.e., $\beta^{(t)} = 0$) and generalization steps at each iteration. In other words, $\mathcal{S}^{(t)} = \mathcal{S}$ and $\mathcal{A}^{(t)} = \mathcal{A}$, for all $t = 1, \ldots, T$, and $\Omega^{(t)} = \{(s,a) \in \mathcal{S}\times\mathcal{A} : s \in \mathcal{S}^\sharp \text{ or } a \in \mathcal{A}^\sharp\}$. Suppose that the optimal matrix $Q^*(\mathcal{S},\mathcal{A})$ is rank-$r$. We then have the following guarantees:

**Corollary 3.** *Consider the rank-r setting with finite* $\mathcal{S}$ *and finite* $\mathcal{A}$*. Suppose that we run the RL algorithm (cf. Section 3.3) with the Matrix Estimation method described in Section 3.5.2. If* $\gamma \le \frac{1}{2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)}$*, then the following statements hold.*

1. *For any* $\delta > 0$*, we have*

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(t)}(s,a) - Q^*(s,a)\right| \le \left(2c(r;\mathcal{S}^\sharp,\mathcal{A}^\sharp)\gamma\right)^t V_{\max}, \quad \text{for all } t = 1,\ldots,T$$

*with probability at least* $1-\delta$ *by choosing algorithmic parameters* $\beta^{(t)}, N^{(t)}$ *appropriately.*

2. *Further, given* $\varepsilon > 0$*, it suffices to set* $T = \Theta(\log \frac{1}{\varepsilon})$ *and use* $\tilde{O}(\frac{\max(|\mathcal{S}|,|\mathcal{A}|)}{\varepsilon^2} \cdot \log \frac{1}{\delta})$ *number of samples to achieve*

$$\mathbb{P}\left( \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \left|Q^{(T)}(s,a) - Q^*(s,a)\right| \le \varepsilon \right) \ge 1-\delta.$$

## ■ B.3 Experimental Setup for Stochastic Control Tasks

In this section, we formalize the detailed settings for the stochastic control tasks we experiment. Following previous work [161, 180], we briefly introduce the background for each task, and then present the system dynamics as well as our simulation setting. For consistency, we follow the dynamics setup in [161, 180], while adding additionally a noise term $\mathcal{N}$ to one dimension of the state dynamics.

**General Setup.** For each task, we first discretize the state space and the the action space into a fine grid and run standard value iteration to obtain a proxy of $Q^*$. Subsequently, when measuring the $\ell_\infty$ error, we take the max (absolute) difference between our estimate

$Q^{(t)}$ and the proxy of $Q^*$ over this fine grid. For the mean error, we use the average of the (absolute) difference over this grid. For anchor states and actions, we simply select $r$ states and $r$ actions that are well separated in their respective space. To do so, we divide the space uniformly into $r$ parts and then select a state/action from each part randomly. We use $r = 10$ in all experiments. In terms of the comparison with different Matrix Estimation methods, we note that as mentioned, the sampling procedure is different: traditional methods often work by independently sampling each entry with some fixed probability $p$, while our method explores a few entire rows and columns. We hence control all the ME methods to have the same number of observations (i.e., same size of the exploration set $\Omega^{(t)}$ as ours) at each iteration, but switch to independent sampling for the traditional methods.

**Inverted Pendulum.** In this control task, we aim to balance an inverted pendulum on the equilibrium position, i.e., the upright position [161]. The angle and the angular speed tuple, $(\theta, \dot{\theta})$, describes the system dynamics, which is formulated as follows [158]:

$$\begin{aligned}\theta &:= \theta + \dot{\theta}\ \tau,\\ \dot{\theta} &:= \dot{\theta} + \Big(\sin\theta - \dot{\theta} + u\Big)\ \tau + \mathcal{N}(\mu, \sigma^2),\end{aligned}$$

where $\tau$ is the time interval between decisions, $u$ denotes the input torque on the pendulum, and $\mathcal{N}$ refers to the noise term we add with mean $\mu$ and variance $\sigma^2$. We formulate the reward function to stabilize the pendulum on an upright pendulum as follows:

$$r(\theta, u) = -0.1u^2 + \exp(\cos\theta - 1).$$

In the simulation, we limit the input torque to $[-1, 1]$ and set $\tau = 0.3$, $\mu = 0$, and $\sigma = 0.1$. We discretize each dimension of the state space into 50 values and action space into 1000 values, which form a discretized matrix of the optimal $Q$-value with dimension $2500 \times 1000$.

**Mountain Car.** The Mountain Car problem aims to drive an under-powered car up to a hill [158]. We use the position and the velocity of the car, $(x, \dot{x})$, to describe the physical dynamics of the system. Denote by $\mathcal{N}$ the noise term added and $u$ the acceleration input on the car, we can express the system dynamics as

$$\begin{aligned}x &:= x + \dot{x} + \mathcal{N}(\mu, \sigma^2),\\ \dot{x} &:= \dot{x} - 0.0025\cos(3x) + 0.001u.\end{aligned}$$

We define a reward function that encourages the car to drive up to the top of the hill at

$x_0 = 0.5$:

$$r(x) = \begin{cases} 10, & x \geq x_0, \\ -1, & \text{else.} \end{cases}$$

We follow the standard setting [180] to limit the input to $u \in [-1, 1]$. We choose $\mu = 0$ and $\sigma = 1e^{-3}$. Similarly, the whole state space is discretized into 2500 values and the action space is discretized into 1000 values, which translate to a $2500 \times 1000$ discretized matrix of the optimal $Q$-value.

**Double Integrator.** We consider the Double Integrator system [133], where a unit mass brick moves along the $x$-axis on a frictionless surface. The brick is controlled with a horizontal input force $u$, which aims to regulate the brick to $\boldsymbol{x} = [0, 0]^T$ [161]. Similarly, we use the position and the velocity $(x, \dot{x})$ of the brick to describe the physical dynamics:

$$\begin{aligned} x &:= x + \dot{x}\ \tau + \mathcal{N}(\mu, \sigma^2), \\ \dot{x} &:= \dot{x} + u\ \tau, \end{aligned}$$

where $\mathcal{N}$ is the noise term added. Following [161], we define the reward function using the quadratic cost formulation:

$$r(x, \dot{x}) = -\frac{1}{2}\left(x^2 + \dot{x}^2\right).$$

The input torque is limited to $u \in [-1, 1]$. We again set $\tau = 0.1$, $\mu = 0$, and $\sigma = 0.1$. Similar to the previous tasks, we obtain a discretization of size $2500 \times 1000$ for the optimal $Q$-value, with state space discretized into 2500 values and action space discretized into 1000 values.

**Cart-Pole.** Besides simple tasks with small state dimensions, we consider the harder Cart-Pole problem with a 4-dimensional state space [18]. The problem consists of a pole attached to a cart moving on a frictionless track, aiming to stabilize the pole at the upright stable position. The cart is controlled by a limited force that can be applied to both sides of the cart. To describe the physical dynamics of the Cart-Pole system, we use a 4-element tuple $(\theta, \dot{\theta}, x, \dot{x})$, corresponding to the angle of the pole, the angular speed of the pole, the position of the cart, and the speed of the cart. The dynamics can be expressed as follows:

$$\begin{aligned} \ddot{\theta} &:= \frac{g \sin\theta - \frac{u + ml\dot{\theta}^2 \sin\theta}{m_c + m} \cos\theta}{l\left(\frac{4}{3} - \frac{m\cos^2\theta}{m_c + m}\right)}, \\ \ddot{x} &:= \frac{u + ml\left(\dot{\theta}^2 \sin\theta - \ddot{\theta}\cos\theta\right)}{m_c + m}, \\ \theta &:= \theta + \dot{\theta}\ \tau, \\ \dot{\theta} &:= \dot{\theta} + \ddot{\theta}\ \tau + \mathcal{N}(\mu, \sigma^2), \end{aligned}$$

$$x := x + \dot{x}\ \tau,$$
$$\dot{x} := \dot{x} + \ddot{x}\ \tau,$$

where $u \in [-10, 10]$ denotes the input applied to the cart, $\mathcal{N}$ with $\mu = 0$ and $\sigma = 0.1$ denotes the noise term, $m_c = 1kg$ denotes the mass of the cart, $m = 0.1kg$ denotes the mass of the pole, and $g = 9.8m/s^2$ corresponds to the gravitational acceleration.

We define the reward function similar to Inverted Pendulum that tries to stabilize the pole in the upright position:

$$r(\theta) = \cos^4(15\theta).$$

In the simulation, we discretize each dimension of the state space into 10 values and action space into 1000 values, which form a descritized optimal $Q$-value matrix of dimension $10000 \times 1000$.

**Acrobot.** Finally, we present the Acrobot swinging up task [161]. The Acrobot is an underactuated two-link robotic arm in the vertical plane (i.e., a two-link pendulum), with only an actuator on the second joint. The goal is to stabilize the Acrobot at the upright position. The equations of motion for the Acrobot can be derived using the method of Lagrange [161]. The physical dynamics of the system is described by the angle and the angular speed of both links, i.e., $(\theta_1, \dot{\theta_1}, \theta_2, \dot{\theta_2})$. Denote by $\tau$ the time interval, $u$ the input force on the second joint, and $\mathcal{N}$ the noise term added, the dynamics of Acrobot can be derived as:

$$\begin{aligned}
D_1 &:= m_1\left(l_1^2 + l_{c1}^2\right) + m_2\left(l_1^2 + l_2^2 + l_{c2}^2 + 2l_1 l_{c2}\cos\theta_2\right),\\
D_2 &:= m_2\left(l_2^2 + l_{c2}^2 + l_1 l_{c2}\cos\theta_2\right),\\
\phi_2 &:= m_2 l_{c2} g\sin\left(\theta_1 + \theta_2\right),\\
\phi_1 &:= -m_2 l_1 l_{c2}\dot{\theta_2}\left(\dot{\theta_2} + 2\dot{\theta_1}\right)\sin\theta_2 + \left(m_1 l_{c1} + m_2 l_1\right) g\sin\theta_1 + \phi_2,\\
\ddot{\theta_2} &:= \frac{u + \frac{D_2}{D_1}\phi_1 - m_2 l_1 l_{c2}\dot{\theta_1}^2\sin\theta_2 - \phi_2}{m_2(l_2^2 + l_{c2}^2) - \frac{D_2^2}{D_1}},\\
\ddot{\theta_1} &:= -\frac{D_2\ddot{\theta_2} + \phi_1}{D_1},\\
\theta_1 &:= \theta_1 + \dot{\theta_1}\ \tau,\\
\dot{\theta_1} &:= \dot{\theta_1} + \ddot{\theta_1}\ \tau + \mathcal{N}(\mu, \sigma^2),\\
\theta_2 &:= \theta_2 + \dot{\theta_2}\ \tau,\\
\dot{\theta_2} &:= \dot{\theta_2} + \ddot{\theta_2}\ \tau,
\end{aligned}$$

where $l_1 = l_2 = 1m$ are the length of the two links, $l_{c1} = l_{c2} = 0.5m$ denote position of the

center of mass of both links, $m_1 = m_2 = 1kg$ denote the mass of two links, and $g = 9.8m/s^2$ denotes the gravitational acceleration. $u$ corresponds to the input force applied, which is limited to $u \in [-10, 10]$.

Similar to the Inverted Pendulum, we define the reward function that favors the Acrobot to stabilize at the upright point $x = [\pi, 0, 0, 0]^T$:

$$r(\theta, u) = \exp(-\cos\theta_1 - 1) + \exp(-\cos(\theta_1 + \theta_2) - 1).$$

Since the state space of Acrobot is also 4-dimensional, we again discretize each dimension of the state space into 10 values and action space into 1000 values. This leads to a discretized optimal $Q$-value matrix with dimension $10000 \times 1000$.

## ■ B.4 Additional Results on Stochastic Control Tasks

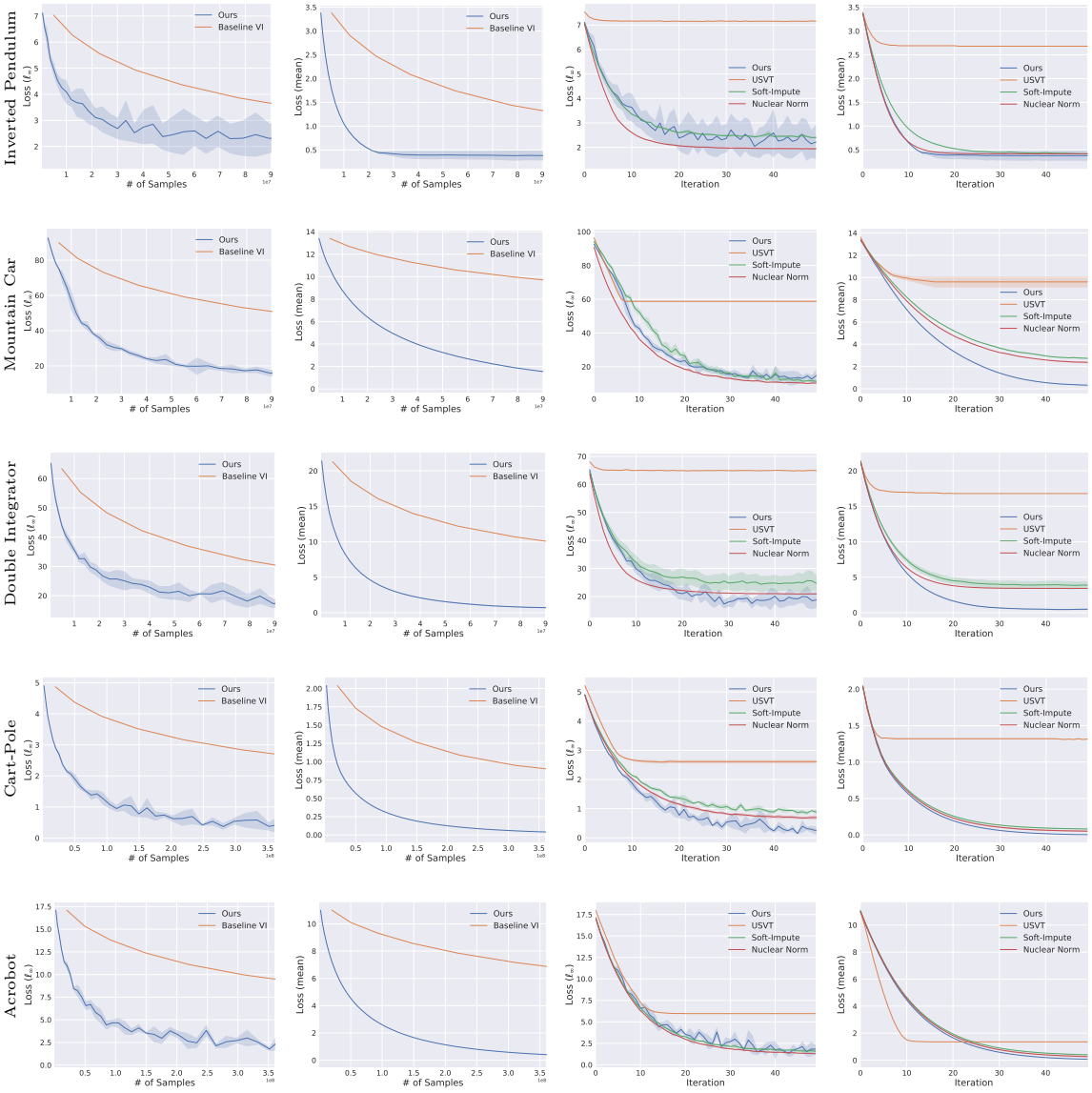
In this section, we provide additional empirical results supporting our theories. These include (1) plots for sample complexity and error guarantees, and visualizations of the learned policies on all of the 5 tasks; (2) runtime comparisons for the various ME methods used in our experiments; (3) results on Inverted Pendulum for a smaller discounting factor.

### ■ B.4.1 Empirical Evaluation on All Control Tasks

**Summary of Empirical Results.** We first remark that the conclusion remains the same as in the main chapter (cf. Section 3.7). Using our low-rank algorithm with the proposed ME method, the sample complexity is significantly improved as compared with the baseline. For the error guarantees, our ME method is very competitive, both in $\ell_\infty$ error and mean error. We again note that our simple method is much more efficient in terms of computational complexity, compared to other ME methods based on solving optimization problems. Finally, the visualization of policies demonstrates that the learned policy, obtained from the output $Q^{(T)}$, is often very close to the policy obtained from $Q^*$. As a result, this leads to the desired behavior in terms of performance metrics as summarized in Table 3.3 of Chapter 3. Overall, these consistent results across various stochastic control tasks confirm the efficacy of our generic low-rank algorithm.

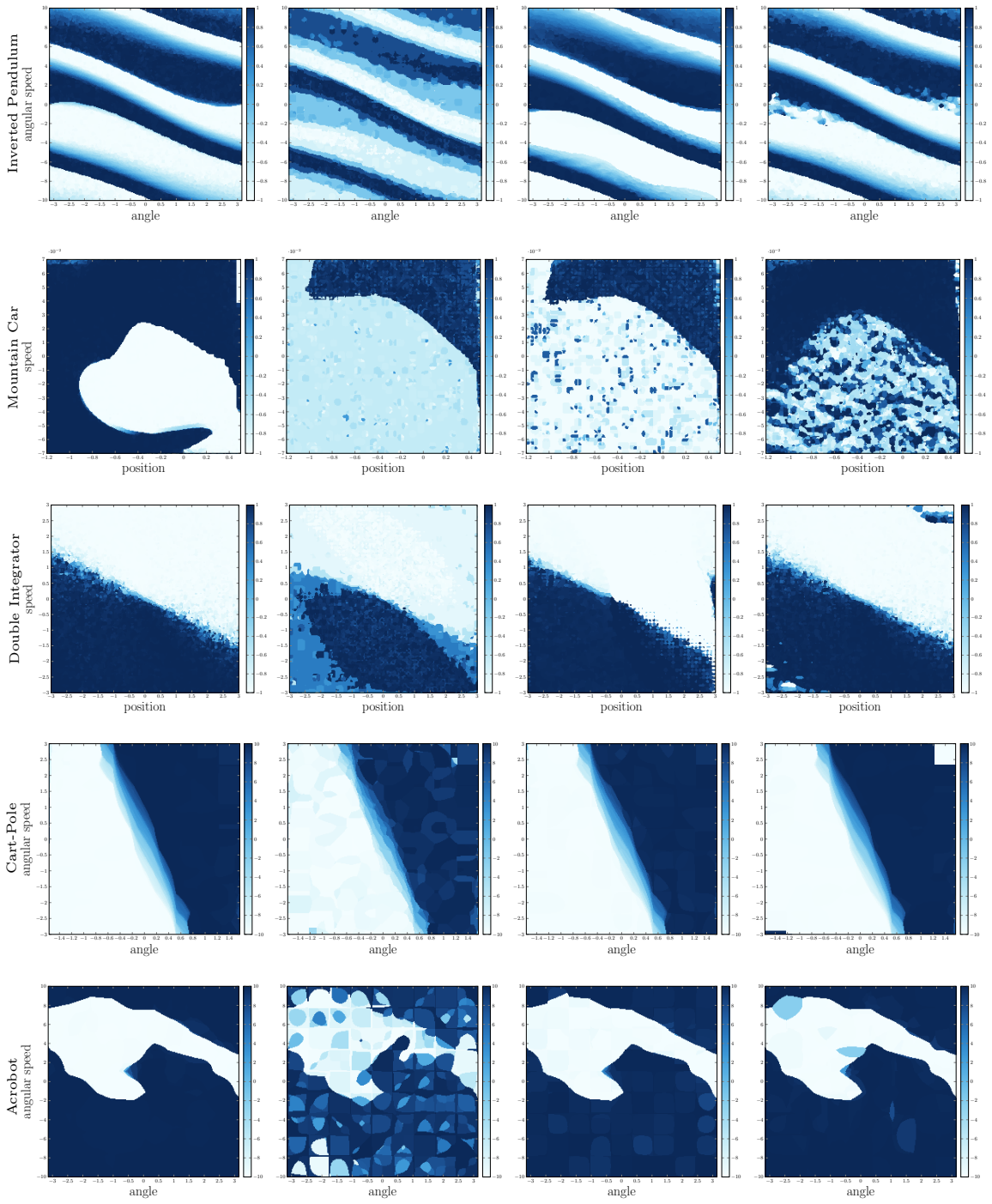
**Sample Complexity and Error Guarantees.** Figure B.1 presents the results with the first two columns showing the sample complexity improvements and the last two columns demonstrating the error guarantees.

**Policy Visualizations.** We visualize the learned policies in Figure B.2 with the first column showing the optimal policy and the other columns displaying policies learned based on our low-rank RL algorithm but with different ME methods.




(a) Sample Complexity (b) Sample Complexity (c) $\ell_\infty$ Errors (d) Mean Errors

**Figure B.1.** Sample complexity and error guarantees for 5 stochastic control tasks. From top to bottom, the rows show empirical results on Inverted Pendulum, Mountain Car, Double Integrator, Cart-Pole and Acrobot, respectively. In columns (a) and (b), we show the improved sample complexity for achieving different levels of $\ell_\infty$ error and mean error, respectively. In columns (c) and (d), we compare the $\ell_\infty$ error and the mean error for various ME methods. Results are averaged across 5 runs for each method.




(a) Optimal Policy (b) Soft-Impute (c) Nuclear Norm (d) Ours

**Figure B.2.** Policy visualizations of different methods for 5 stochastic control tasks. From top to bottom, the rows show empirical results on Inverted Pendulum, Mountain Car, Double Integrator, Cart-Pole and Acrobot, respectively. The policies are obtained from the corresponding output $Q^{(T)}$ by taking $\arg\max_{a\in\mathcal{A}} Q^{(T)}(s,a)$ at each state $s$. Recall the for both Cart-Pole and Acrobot, the state space is 4-dimensional. We hence visualize a 2-dimensional slice in the figures above.

## ■ B.4.2 Comparison on Runtime of Different ME Methods

Nuclear norm minimization is known to be computationally expensive for large matrices. Here, we provide a preliminary result on the runtime of different ME methods in our experiments, demonstrating the computational gain of our approach beyond its theoretical guarantees. Specifically, we calculate the average runtime for one iteration using different ME methods on the Inverted Pendulum task with a 2500×1000 matrix. We leave other hyper-parameters unchanged, and perform 5 runs for each method. As Table B.1 reports, the nuclear norm minimization is, as expected, computationally most expensive; in contrast, our method is about 40x faster, confirming the computational efficiency of our approach.

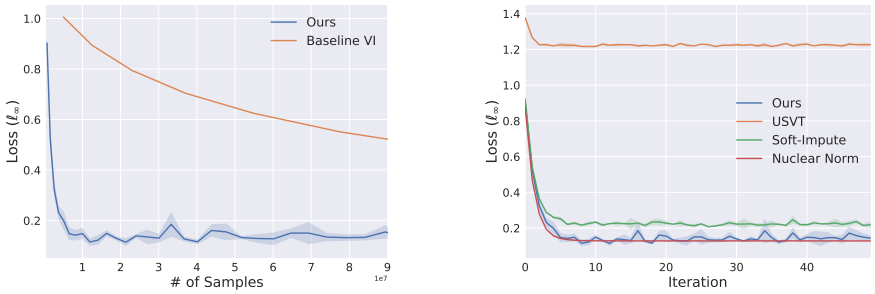
**Table B.1.** The runtime comparison of different ME methods for one iteration on the Inverted Pendulum task. Results are averaged across 5 runs for each method.

| ME Method | Soft-Impute | Nuclear norm | Ours |
|---|---|---|---|
| Runtime (s) | 41.5 ± 1.7 | 76.3 ± 8.2 | 1.9 ± .6 |

## ■ B.4.3 Additional Study on the Discounting Factor $\gamma$

Throughout the empirical study, we follow the literature [161, 180] to use a large discounting factor $\gamma$ (i.e., 0.9) on several real control tasks. We have demonstrated that the proposed low-rank algorithm can perform well on those settings, confirming the efficacy of our method. Just as a final proof of concept for our theoretical guarantees, we provide in this section an ablation study on the $\ell_\infty$ error with smaller value of $\gamma$. We choose $\gamma = 0.5$ on the Inverted Pendulum control task (note that this would affect the reward design and change the original task). The experiment is only meant to further validate our guarantees.

We show the sample complexity as well as the $\ell_\infty$ errors in Figure B.3. As expected, with a smaller $\gamma$, the convergence is faster. Again, the overall conclusion is consistent with the previous experiments: significant gains on sample complexity are achieved by our efficient algorithm, and the performance of our simple ME method is competitive.




(a) Sample Complexity, $\gamma = 0.5$ (b) $\ell_\infty$ Errors, $\gamma = 0.5$

**Figure B.3.** Empirical results on the Inverted Pendulum control task with $\gamma = 0.5$. We show the improved sample complexity in (a) and compare the $\ell_\infty$ error for various ME methods in (b).

# Bibliography


[1] Yasin Abbasi-Yadkori and Csaba Szepesvári. Regret bounds for the adaptive control of linear quadratic systems. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 1–26, 2011.

[2] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 22–31. JMLR. org, 2017.

[3] Alekh Agarwal, Sham Kakade, Akshay Krishnamurthy, and Wen Sun. Flambe: Structural complexity and representation learning of low rank mdps. *arXiv preprint arXiv:2006.10814*, 2020.

[4] Anish Agarwal, Abdullah Alomar, Varkey Alumootil, Devavrat Shah, Dennis Shen, Zhi Xu, and Cindy Yang. Persim: Data-efficient offline reinforcement learning with heterogeneous agents via personalized simulators. *arXiv preprint arXiv:2102.06961*, 2021.

[5] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pages 104–114. PMLR, 2020.

[6] Rajeev Agrawal. Sample mean based index policies by o (log n) regret for the multi-armed bandit problem. *Advances in Applied Probability*, 27(4):1054–1078, 1995.

[7] Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

[8] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *arXiv preprint arXiv:1707.01495*, 2017.

[9] András Antos, Csaba Szepesvári, and Rémi Munos. Fitted q-iteration in continuous action-space mdps. In *Advances in neural information processing systems*, pages 9–16, 2008.

[10] Sanjeev Arora, Rong Ge, and Ankur Moitra. Learning topic models–going beyond svd. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 1–10. IEEE, 2012.

[11] Anil Aswani, Humberto Gonzalez, S Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, 2013.

[12] Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. Exploration–exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410(19):1876–1902, 2009.

[13] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[14] David Auger, Adrien Couetoux, and Olivier Teytaud. Continuous upper confidence trees with polynomial exploration–consistency. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 194–209. Springer, 2013.

[15] Mohammad Gheshlaghi Azar, Rémi Munos, and Hilbert J Kappen. Minimax pac bounds on the sample complexity of reinforcement learning with a generative model. *Machine learning*, 91(3):325–349, 2013.

[16] Kamyar Azizzadenesheli, Brandon Yang, Weitang Liu, Emma Brunskill, Zachary C. Lipton, and Animashree Anandkumar. Sample-efficient deep RL with generative adversarial tree search. *CoRR*, abs/1806.05780, 2018.

[17] Peter Bartlett, Victor Gabillon, Jennifer Healey, and Michal Valko. Scale-free adaptive planning for deterministic dynamics & discounted rewards. In *International Conference on Machine Learning*, pages 495–504, 2019.

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

[19] Felix Berkenkamp, Angela P Schoellig, and Andreas Krause. Safe controller optimization for quadrotors with gaussian processes. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 491–496. IEEE, 2016.

[20] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Advances in neural information processing systems*, pages 908–918, 2017.

[21] D. Bertsekas. Convergence of discretization procedures in dynamic programming. *IEEE Transactions on Automatic Control*, 20(3):415–419, 1975.

[22] Dimitri P Bertsekas and John N Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, 1996.

[23] D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2017.

[24] Rajendra Bhatia. *Notes on functional analysis*, volume 50. Springer, 2009.

[25] Carlos F Bispo. Single server scheduling problem: Optimal policy for convex costs depends on arrival rates. In *Proc. Multidisciplinary Int. Conf. on Scheduling: Theory and Applications (MISTA 2011)*, pages 275–296, 2011.

[26] Cameron B. Browne, Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[27] Sébastien Bubeck, Nicolò Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[28] C Buyukkoc, P Varaiya, and Jean Walrand. The c$\mu$ rule revisited. *Advances in applied probability*, 17(1):237–238, 1985.

[29] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer science & business media, 2013.

[30] Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

[31] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717, 2009.

[32] Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

[33] Hyeong Soo Chang, Michael C. Fu, Jiaqiao Hu, and Steven I Marcus. An adaptive sampling algorithm for solving markov decision processes. *Operations Research*, 53(1):126–139, 2005.

[34] Sourav Chatterjee et al. Matrix estimation by universal singular value thresholding. *The Annals of Statistics*, 43(1):177–214, 2015.

[35] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. Top-k off-policy correction for a reinforce recommender system. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 456–464, 2019.

[36] Steven Chen, Kelsey Saulnier, Nikolay Atanasov, Daniel D Lee, Vijay Kumar, George J Pappas, and Manfred Morari. Approximating explicit model predictive control using constrained neural networks. In *2018 Annual American Control Conference (ACC)*, pages 1520–1527. IEEE, 2018.

[37] Yudong Chen and Yuejie Chi. Harnessing structures in big data via guaranteed low-rank matrix estimation. *arXiv preprint arXiv:1802.08397*, 2018.

[38] Yudong Chen and Martin J Wainwright. Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025*, 2015.

[39] Yuxin Chen, Yuejie Chi, Jianqing Fan, Cong Ma, and Yuling Yan. Noisy matrix completion: Understanding statistical guarantees for convex relaxation via nonconvex optimization. *arXiv preprint arXiv:1902.07698*, 2019.

[40] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 8092–8101, 2018.

[41] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *arXiv preprint arXiv:1805.12114*, 2018.

[42] Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-based reinforcement learning via meta-policy optimization. In *Conference on Robot Learning*, pages 617–629. PMLR, 2018.

[43] John B Conway. *A course in functional analysis*, volume 96. Springer, 2019.

[44] Pierre-Arnaud Coquelin and Rémi Munos. Bandit algorithms for tree search. *arXiv preprint cs/0703062*, 2007.

[45] Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pages 72–83. Springer, 2006.

[46] J. G. Dai and B. Prabhakar. The throughput of data switches with and without speedup. In *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, volume 2, pages 556–564 vol.2, 2000.

[47] JG Dai and Mark Gluzman. Queueing network controls via deep reinforcement learning. *arXiv preprint arXiv:2008.01644*, 2020.

[48] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

[49] Mark A Davenport and Justin Romberg. An overview of low-rank matrix recovery from incomplete observations. *arXiv preprint arXiv:1601.06422*, 2016.

[50] Sarah Dean, Stephen Tu, Nikolai Matni, and Benjamin Recht. Safely learning to control the constrained linear quadratic regulator. In *2019 American Control Conference (ACC)*, pages 5582–5588. IEEE, 2019.

[51] Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472. Citeseer, 2011.

[52] Lijun Ding and Yudong Chen. Leave-one-out approach for matrix completion: Primal and dual analysis. *IEEE Transactions on Information Theory*, 2020.

[53] Yaqi Duan, Tracy Ke, and Mengdi Wang. State aggregation learning from markov transition data. In *Advances in Neural Information Processing Systems*, pages 4488–4497, 2019.

[54] F. Dufour and T. Prieto-Rumeau. Finite linear programming approximations of constrained discounted Markov decision processes. *SIAM Journal on Control and Optimization*, 51(2):1298–1324, 2013.

[55] François Dufour and Tomás Prieto-Rumeau. Approximation of markov decision processes with general state space. *Journal of Mathematical Analysis and applications*, 388(2):1254–1267, 2012.

[56] Yonathan Efroni, Gal Dalal, Bruno Scherrer, and Shie Mannor. Multiple-step greedy policies in approximate and online reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 5244–5253, 2018.

[57] Atilla Eryilmaz and Rayadurgam Srikant. Asymptotically tight steady-state queue length bounds implied by drift conditions. *Queueing Systems*, 72(3-4):311–359, 2012.

[58] E. Even-Dar and Y. Mansour. Learning rates for Q-learning. *JMLR*, 5, December 2004.

[59] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

[60] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019.

[61] Carlos E Garcia, David M Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989.

[62] Javier Garcia and Fernando Fernández. Safe exploration of state and action spaces in reinforcement learning. *Journal of Artificial Intelligence Research*, 45:515–564, 2012.

[63] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[64] Rong Ge, Jason D Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pages 2973–2981, 2016.

[65] Peter W Glynn, Assaf Zeevi, et al. Bounding stationary expectations of markov processes. In *Markov processes and related topics: a Festschrift for Thomas G. Kurtz*, pages 195–214. Institute of Mathematical Statistics, 2008.

[66] Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, Bernhard Schölkopf, and Sergey Levine. Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning. *arXiv preprint arXiv:1706.00387*, 2017.

[67] Xiaoxiao Guo, Satinder Singh, Honglak Lee, Richard L Lewis, and Xiaoshi Wang. Deep learning for real-time atari game play using offline monte-carlo tree search planning. In *Advances in neural information processing systems*, pages 3338–3346, 2014.

[68] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[69] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. PMLR, 2018.

[70] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pages 2555–2565. PMLR, 2019.

[71] Bruce Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied probability*, 14(3):502–525, 1982.

[72] Shlomo Halfin and Ward Whitt. Heavy-traffic limits for queues with many exponential servers. *Operations research*, 29(3):567–588, 1981.

[73] Alexander Hans, Daniel Schneegaß, Anton Maximilian Schäfer, and Steffen Udluft. Safe exploration for reinforcement learning. In *ESANN*, pages 143–148, 2008.

[74] J Michael Harrison. The diffusion approximation for tandem queues in heavy traffic. *Advances in Applied Probability*, 10(4):886–905, 1978.

[75] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[76] W Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of Americal Statistics Association*, 58, 1963.

[77] Jean-Francois Hren and Rémi Munos. Optimistic planning of deterministic systems. In *European Workshop on Reinforcement Learning*, pages 151–164. Springer, 2008.

[78] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *arXiv preprint arXiv:1906.08253*, 2019.

[79] Daniel R. Jiang, Emmanuel Ekwedike, and Han Liu. Feedback-based tree search for reinforcement learning. In *International conference on machine learning*, 2018.

[80] Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual decision processes with low bellman rank are pac-learnable. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1704–1713, 2017.

[81] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143. PMLR, 2020.

[82] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[83] 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.

[84] Sham Kakade. *On the sample complexity of reinforcement learning*. PhD thesis, 2003.

[85] Sham M Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14, 2001.

[86] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

[87] Emilie Kaufmann and Wouter M Koolen. Monte-carlo tree search by best arm identification. In *Advances in Neural Information Processing Systems*, pages 4897–4906, 2017.

[88] Michael Kearns, Yishay Mansour, and Andrew Y. Ng. A sparse sampling algorithm for near-optimal planning in large markov decision processes. *Machine learning*, 49(2-3):193–208, 2002.

[89] Michael Kearns, Yishay Mansour, and Satinder Singh. Fast planning in stochastic games. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 309–316, 2000.

[90] Michael Kearns and Satinder Singh. Finite-sample convergence rates for q-learning and indirect algorithms. In *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II*, pages 996–1002, Cambridge, MA, USA, 1999. MIT Press.

[91] Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8248–8254. IEEE, 2019.

[92] Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *IEEE transactions on information theory*, 56(6):2980–2998, 2010.

[93] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

[94] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer, 2006.

[95] Levente Kocsis, Csaba Szepesvári, and Jan Willemson. Improved monte-carlo search. *Univ. Tartu, Estonia, Tech. Rep*, 2006.

[96] Torsten Koller, Felix Berkenkamp, Matteo Turchetta, and Andreas Krause. Learning-based model predictive control for safe exploration. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6059–6066. IEEE, 2018.

[97] Vladimir Koltchinskii, Karim Lounici, Alexandre B Tsybakov, et al. Nuclear-norm penalization and optimal rates for noisy low-rank matrix completion. *The Annals of Statistics*, 39(5):2302–2329, 2011.

[98] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014. Citeseer, 2000.

[99] Subhashini Krishnasamy, Ari Arapostathis, Ramesh Johari, and Sanjay Shakkottai. On learning the c$\mu$ rule in single and parallel server networks. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 153–154. IEEE, 2018.

[100] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11784–11794, 2019.

[101] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

[102] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. *arXiv preprint arXiv:1802.10592*, 2018.

[103] Harold Kushner. *Heavy traffic analysis of controlled queueing and communication networks*, volume 47. Springer Science & Business Media, 2013.

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

[105] Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

[106] Alessandro Lazaric, Marcello Restelli, and Andrea Bonarini. Reinforcement learning in continuous action spaces through sequential monte carlo methods. In *Advances in neural information processing systems*, pages 833–840, 2008.

[107] Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 5757–5766. PMLR, 2020.

[108] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

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

[110] 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. 2016.

[111] Bai Liu, Qiaomin Xie, and Eytan Modiano. Reinforcement learning for optimal control of queueing systems. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 663–670. IEEE, 2019.

[112] Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Provably good batch reinforcement learning without great exploration. *arXiv preprint arXiv:2007.08202*, 2020.

[113] Yuping Luo, Huazhe Xu, Yuanzhi Li, Yuandong Tian, Trevor Darrell, and Tengyu Ma. Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. *arXiv preprint arXiv:1807.03858*, 2018.

[114] Hamid Reza Maei, Csaba Szepesvári, Shalabh Bhatnagar, and Richard S Sutton. Toward off-policy learning control with function approximation. In *ICML*, 2010.

[115] Hongzi Mao, Malte Schwarzkopf, Shaileshh Bojja Venkatakrishnan, Zili Meng, and Mohammad Alizadeh. Learning scheduling algorithms for data processing clusters. In *Proceedings of the ACM Special Interest Group on Data Communication*, pages 270–288. ACM, 2019.

[116] Hongzi Mao, Shaileshh Bojja Venkatakrishnan, Malte Schwarzkopf, and Mohammad Alizadeh. Variance reduction for reinforcement learning in input-driven environments. *arXiv preprint arXiv:1807.02264*, 2018.

[117] Weichao Mao, Kaiqing Zhang, Qiaomin Xie, and Tamer Başar. Poly-hoot: Monte-carlo planning in continuous space mdps with non-asymptotic analysis. *arXiv preprint arXiv:2006.04672*, 2020.

[118] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research*, 11(Aug):2287–2322, 2010.

[119] Francisco S Melo, Sean P Meyn, and M Isabel Ribeiro. An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th international conference on Machine learning*, pages 664–671, 2008.

[120] Jean-François Mertens, Ester Samuel-Cahn, and Shmuel Zamir. Necessary and sufficient conditions for recurrence and transience of markov chains, in terms of inequalities. *Journal of Applied Probability*, 15(4):848–851, 1978.

[121] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[122] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[123] Ciamac C Moallemi, Sunil Kumar, and Benjamin Van Roy. Approximate and data-driven dynamic programming for queueing networks. *Submitted for publication*, 2008.

[124] Rémi Munos et al. From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. *Foundations and Trends® in Machine Learning*, 7(1):1–129, 2014.

[125] Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(May):815–857, 2008.

[126] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[127] Xinkun Nie, Emma Brunskill, and Stefan Wager. Learning when-to-treat policies. *Journal of the American Statistical Association*, pages 1–18, 2020.

[128] David C Parkes, Dimah Yanovsky, and Satinder P Singh. Approximately efficient online mechanism design. In *Advances in Neural Information Processing Systems*, pages 1049–1056, 2005.

[129] Ronald Parr, Lihong Li, Gavin Taylor, Christopher Painter-Wakefield, and Michael L Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 752–759, 2008.

[130] Martin Pecka and Tomas Svoboda. Safe exploration techniques for reinforcement learning–an overview. In *International Workshop on Modelling and Simulation for Autonomous Systems*, pages 357–375. Springer, 2014.

[131] Theodore J Perkins and Andrew G Barto. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research*, 3(Dec):803–832, 2002.

[132] Benjamin Recht, Maryam Fazel, and Pablo A Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[133] Wei Ren and Randal W Beard. Consensus algorithms for double-integrator dynamics. *Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications*, pages 77–104, 2008.

[134] Benjamin V Roy and Daniela D Farias. Approximate linear programming for average-cost dynamic programming. In *Advances in neural information processing systems*, pages 1619–1626, 2003.

[135] Dorsa Sadigh and Ashish Kapoor. Safe control under uncertainty with probabilistic signal temporal logic. In *Proceedings of Robotics: Science and Systems*, 2016.

[136] Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

[137] Antoine Salomon and Jean-Yves Audibert. Deviations of stochastic bandit regret. In *International Conference on Algorithmic Learning Theory*, pages 159–173. Springer, 2011.

[138] Maarten P. D. Schadd, Mark H. M. Winands, H. Jaap van den Herik, Guillaume M. J. B. Chaslot, and Jos W. H. M. Uiterwijk. Single-player monte-carlo tree search. In H. Jaap van den Herik, Xinhe Xu, Zongmin Ma, and Mark H. M. Winands, editors, *Computers and Games*, pages 1–12, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[139] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[140] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

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

[142] Devavrat Shah, Dogyoon Song, Zhi Xu, and Yuzhe Yang. Sample efficient reinforcement learning via low-rank matrix estimation. *arXiv preprint arXiv:2006.06135*, 2020.

[143] Devavrat Shah and Qiaomin Xie. Q-learning with nearest neighbors. In *Advances in Neural Information Processing Systems*, pages 3111–3121, 2018.

[144] Devavrat Shah, Qiaomin Xie, and Zhi Xu. Non-asymptotic analysis of monte carlo tree search. In *Abstracts of the 2020 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems*, pages 31–32, 2020.

[145] Devavrat Shah, Qiaomin Xie, and Zhi Xu. Stable reinforcement learning with unbounded state space. *arXiv preprint arXiv:2006.04353*, 2020.

[146] Aaron Sidford, Mengdi Wang, Xian Wu, Lin Yang, and Yinyu Ye. Near-optimal time and sample complexities for solving markov decision processes with a generative model. In *Advances in Neural Information Processing Systems*, pages 5186–5196, 2018.

[147] Aaron Sidford, Mengdi Wang, Xian Wu, and Yinyu Ye. Variance reduced value iteration and faster algorithms for solving markov decision processes. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 770–787. SIAM, 2018.

[148] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[149] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

[150] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[151] R Srikant and Lei Ying. Finite-time error bounds for linear stochastic approximation and td learning. *arXiv preprint arXiv:1902.00923*, 2019.

[152] Rayadurgam Srikant and Lei Ying. *Communication networks: an optimization, control, and stochastic networks perspective*. Cambridge University Press, 2013.

[153] Gilbert W Stewart. On the perturbation of pseudo-inverses, projections and linear least squares problems. *SIAM review*, 19(4):634–662, 1977.

[154] Charles J. Stone. Optimal global rates of convergence for nonparametric regression. *The Annals of Statistics*, pages 1040–1053, 1982.

[155] Alexander L. Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888. ACM, 2006.

[156] Nathan R. Sturtevant. An analysis of uct in multi-player games. In H. Jaap van den Herik, Xinhe Xu, Zongmin Ma, and Mark H. M. Winands, editors, *Computers and Games*, pages 37–49, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[157] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

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

[159] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miroslav Dudík, John Langford, Damien Jose, and Imed Zitouni. Off-policy evaluation for slate recommendation. *arXiv preprint arXiv:1605.04812*, 2016.

[160] Csaba Szepesvári. Personal communication. January 2019.

[161] Russ Tedrake. Underactuated robotics: Algorithms for walking, running, swimming, flying, and manipulation. *Course Notes for MIT 6.832*, 2020.

[162] Kazuki Teraoka, Kohei Hatano, and Eiji Takimoto. Efficient sampling method for monte carlo tree search problem. *IEICE TRANSACTIONS on Information and Systems*, 97(3):392–398, 2014.

[163] John N Tsitsiklis and Benjamin Van Roy. Analysis of temporal-diffference learning with function approximation. In *Advances in neural information processing systems*, pages 1075–1081, 1997.

[164] Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics. Springer, 2009.

[165] Matteo Turchetta, Felix Berkenkamp, and Andreas Krause. Safe exploration in finite markov decision processes with gaussian processes. In *Advances in Neural Information Processing Systems*, pages 4312–4320, 2016.

[166] Hado Van Hasselt. Reinforcement learning in continuous state and action spaces. In *Reinforcement learning*, pages 207–251. Springer, 2012.

[167] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *AAAI*, volume 2, page 5. Phoenix, AZ, 2016.

[168] Julia Vinogradska, Bastian Bischoff, Duy Nguyen-Tuong, Anne Romer, Henner Schmidt, and Jan Peters. Stability of controllers for gaussian process forward models. In *International Conference on Machine Learning*, pages 545–554, 2016.

[169] Kim P Wabersich and Melanie N Zeilinger. Safe exploration of nonlinear dynamical systems: A predictive safety filter for reinforcement learning. *arXiv preprint arXiv:1812.05506*, 2018.

[170] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

[171] Lu Wang, Wei Zhang, Xiaofeng He, and Hongyuan Zha. Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2447–2456, 2018.

[172] Tingwu Wang and Jimmy Ba. Exploring model-based planning with policy networks. *arXiv preprint arXiv:1906.08649*, 2019.

[173] Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

[174] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003. PMLR, 2016.

[175] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[176] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pages 5–32. Springer, 1992.

[177] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[178] Lin Yang and Mengdi Wang. Sample-optimal parametric q-learning using linearly additive features. In *International Conference on Machine Learning*, pages 6995–7004. PMLR, 2019.

[179] Lin F Yang and Mengdi Wang. Reinforcement leaning in feature space: Matrix bandit, kernels, and regret bound. *arXiv preprint arXiv:1905.10389*, 2019.

[180] Yuzhe Yang, Guo Zhang, Zhi Xu, and Dina Katabi. Harnessing structures for value-based planning and reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[181] Zhuora Yang, Yuchen Xie, and Zhaoran Wang. A theoretical analysis of deep q-learning. *arXiv preprint arXiv:1901.00137*, 2019.

[182] Ming Yu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. Convergent policy optimization for safe reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 3121–3133, 2019.

[183] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*, 2020.

[184] Shaofeng Zou, Tengyu Xu, and Yingbin Liang. Finite-sample analysis for sarsa with linear function approximation. In *Advances in Neural Information Processing Systems*, pages 8665–8675, 2019.