Finite-sample Analysis of Greedy-GQ with Linear Function Approximation under Markovian Noise

Yue Wang
Electrical Engineering
University at Buffalo
ywang294@buffalo.edu

Shaofeng Zou
Electrical Engineering
University at Buffalo
szou3@buffalo.edu

Abstract

Greedy-GQ is an off-policy two timescale algorithm for optimal control in reinforcement learning [18]. This paper develops the first finite-sample analysis for the Greedy-GQ algorithm with linear function approximation under Markovian noise. Our finite-sample analysis provides theoretical justification for choosing step-sizes for this two timescale algorithm for faster convergence in practice, and suggests a trade-off between the convergence rate and the quality of the obtained policy. Our paper extends the finite-sample analyses of two timescale reinforcement learning algorithms from policy evaluation to optimal control, which is of more practical interest. Specifically, in contrast to existing finite-sample analyses for two timescale methods, e.g., GTD, GTD2 and TDC, where their objective functions are convex, the objective function of the Greedy-GQ algorithm is non-convex. Moreover, the Greedy-GQ algorithm is also not a linear two-timescale stochastic approximation algorithm. Our techniques in this paper provide a general framework for finite-sample analysis of non-convex value-based reinforcement learning algorithms for optimal control.

1 INTRODUCTION

Reinforcement learning (RL) is to find an optimal control policy to interact with a (stochastic) environment so that the accumulated reward is maximized [27]. It finds a wide range of applications in practice, e.g., robotics, computer games and recommendation systems [21, 20, 25, 14].

When the state and action spaces of the RL problem are finite and small, RL algorithms based on the tabular approach, which stores the action-values for each state-action pair, can be applied and usually have convergence guarantee, e.g., Q-learning [32] and SARSA [24]. However, in many RL applications, the state and action spaces are very large or even continuous. Then, the approach of function approximation can be used. Nevertheless, with function approximation in off-policy training, classical RL algorithms may diverge to infinity, e.g., Q-learning, SARSA and TD learning [2, 11].

To address the non-convergence issue in off-policy training, a class of gradient temporal difference (GTD) learning algorithms were developed in [18, 17, 28, 29], including GTD, GTD2, TD with correction term (TDC), and Greedy-GQ. The basic idea is to construct squared objective functions, e.g., mean squared projected Bellman error, and then to perform stochastic gradient descent. To address the double sampling problem in gradient estimation, a weight doubling trick was proposed in [28], which leads to a two timescale update rule. One great advantage of this class of algorithms is that they can be implemented in an online and incremental fashion, which is memory and computationally efficient.

The asymptotic convergence of these two timescale algorithms has been well studied under both i.i.d. and non-i.i.d. settings [28, 29, 18, 34, 4, 5, 13]. Furthermore, the finite-sample analyses of these algorithms are of great practical interest for algorithmic parameter tuning and design of new sample-efficient algorithms. However, these problems remain unsolved until very recently [8, 31, 16, 12, 33]. But, existing finite-sample analyses are only for the GTD, GTD2 and TDC algorithms, which are designed for evaluation of a given policy. The finite-sample analysis for the Greedy-GQ algorithm, which is to directly learn an optimal control policy, is still not understood and will be the focus of this paper.

In this paper, we will develop the finite-sample analysis for the Greedy-GQ algorithm with linear function approximation under Markovian noise. More specifically, we

Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI), PMLR volume 124, 2020.

focus on the general case with a single sample trajectory and non-i.i.d. data. We will develop explicit bounds on the convergence of the Greedy-GQ algorithm and understand its sample complexity as a function of various parameters of the algorithm.

1.1 Summary of Major Challenges and Contributions

The major challenges and our main contributions are summarized as follows.

The objective function of the Greedy-GQ algorithm is the mean squared projected Bellman error (MSPBE). Unlike the objective functions of GTD, GTD2 and TDC, which are convex, the objective function of Greedy-GQ is non-convex since the target policy is also a function of the action-value function approximation (see (9) for the objective function). In this case, the Greedy-GQ algorithm may not be able to converge to the global optimum, and existing analyses for GTD, GTD2 and TDC based on convex optimization theory cannot be directly applied. Moreover, the Greedy-GQ algorithm cannot be viewed as a linear two timescale stochastic approximation due to its non-convexity, and thus existing analyses for linear two timescale stochastic approximation are not applicable. Due to the non-convexity of the objective function, convergence to the global optimum may not be guaranteed. Therefore, we study the convergence of the gradient norm to zero (in an on-average sense, i.e., randomized stochastic gradient method [10]), and we focus on convergence to stationary points. In this paper, we develop a novel methodology for finite-sample analysis of the Greedy-GQ algorithm, which solves reinforcement learning problems from a non-convex optimization perspective. This may be of independent interest for a wide range of reinforcement learning problems with non-convex objective functions.

In this paper, we focus on the most general scenario where there is a single sample trajectory and the data are non-i.i.d.. This non-i.i.d. setting will invalidate the martingale noise assumption commonly used in stochastic approximation (SA) analysis [18, 8, 5]. Our approach is to analyze RL algorithms from a non-convex optimization perspective, and does not require the martingale noise assumption. Thus, our approach has a much broader applicability.

Moreover, the propagation of the stochastic bias in the gradient estimate caused by the Markovian noise in the two timescale updates makes the analysis even more challenging. We develop a comprehensive characterization of the stochastic bias and establish the convergence rate of the Greedy-GQ algorithm under constant step-sizes. More importantly, we develop a novel recursive approach of bounding the bias caused by the tracking error, i.e., the error in the fast timescale update. Specifically, our approach is to recursively plug the obtained bound back into the analysis to tighten the final bound on the bias.

We show that under constant step-sizes, i.e., $\alpha_t = \frac{1}{T^a}$ and $\beta_t = \frac{1}{T^b}$ for $0 \leq t \leq T$, the Greedy-GQ algorithm converges as fast as $\mathcal{O}\left(\frac{1}{T^{1-a}} + \frac{\log T}{T^{\min\{b, a-b\}}}\right)$. We also derive the best choice of a and b so that the above rate is the fastest. Specifically, when $a = \frac{2}{3}$ and $b = \frac{1}{3}$, the Greedy-GQ algorithm converges as fast as $\mathcal{O}\left(\frac{\log T}{T^{\frac{1}{3}}}\right)$. We further characterize the trade-off between the convergence speed and the quality of the obtained policy. Specifically, the algorithm needs more samples to converge if the target policy is more "greedy", e.g., a larger parameter σ in softmax makes the policy more "greedy", and will require more samples to converge. Our experiments also validate this theoretical observation.

1.2 Related Work

In this subsection, we provide an overview of closely related work. Specifically, we here focus on value-based RL algorithms with function approximation. We note that there are many other types of approaches, e.g., policy gradient and fitted value/policy iteration, which are not discussed in this paper.

TD, Q-learning and SARSA with function approximation. TD with linear function approximation was shown to converge asymptotically in [30], and its finite-sample analysis was established in [9, 15, 3, 26] under both i.i.d. and non-i.i.d. settings. Moreover, the finite-sample analysis of TD with over–parameterized neural function approximation was developed in [6]. Q-learning and SARSA with linear function approximation were shown to converge asymptotically under certain conditions [19, 23] and their finite-sample analyses were developed in [35, 7]. Although they may have a faster convergence rate [35, 7], however, these algorithms may diverge under off-policy training, e.g., Baird's counterexample [2]. Different from TD, Q-learning and SARSA, the Greedy-GQ algorithm follows a stochastic gradient descent type update. However, the updates of TD, Q-learning and SARSA do not exactly follow a gradient descent type, since the "gradient" therein is not gradient of any function [18]. The Greedy-GQ algorithm is a two timescale one, and thus requires more involved analysis than these one timescale methods. Moreover, the Greedy-GQ algorithm is essentially a non-convex optimization problem, for which the convergence is in general slower than that of a convex problem.

GTD algorithms. The GTD, GTD2 and TDC algorithms

were shown to converge asymptotically in [29, 28, 34]. Their finite-sample analyses were further developed recently in [8, 31, 16, 12, 33] under i.i.d. and non-i.i.d. settings. The Greedy-GQ algorithm studied in this paper is fundamentally different from the above three algorithms. This is due to the fact that the Greedy-GQ algorithm is for optimal control and its objective function is non-convex; whereas the GTD, GTD2 and TDC algorithms are for policy evaluation, and their objective functions are convex. Therefore, new techniques need to be developed to tackle the non-convexity for the finite-sample analysis for Greedy-GQ. Moreover, general linear two timescale stochastic approximation has also been studied. Although the Greedy-GQ algorithm follows a two timescale update rule, but it is not linear. Furthermore, the general non-linear two timescale stochastic approximation was studied in [5]. However, the Greedy-GQ algorithm under Markovian noise does not satisfy the martingale noise assumption therein. Moreover, our paper uses a non-convex optimization based approach to develop the finite-sample analysis, which is different from the approach used in [5].

2 PRELIMINARIES

2.1 Markov Decision Process

In RL problems, a Markov Decision Process (MDP) is usually used to model the interaction between an agent and a stochastic environment. Specifically, an MDP consists of $(\mathcal{S}, \mathcal{A}, \mathsf{P}, r, \gamma)$, where $\mathcal{S} \subset \mathbb{R}^d$ is the state space, $\mathcal{A}$ is a finite set of actions, and $\gamma \in (0,1)$ is the discount factor. Denote the state at time t by S_t, and the action taken at time t by A_t. Then the measure P denotes the action-dependent transition kernel of the MDP:

$$\mathbb{P}(S_{t+1} \in U | S_t = s, A_t = a) = \int_U \mathsf{P}(dx|s,a), \quad (1)$$

for any measurable set $U \subseteq \mathcal{S}$. The reward at time t is given by $r_t = r(S_t, A_t, S_{t+1})$, which is the reward of taking action A_t at state S_t and transitioning to a new state S_{t+1}. Here $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^+$ is the reward function, and is assumed to be uniformly bounded, i.e.,

$$0 \leq r(s,a,s') \leq r_{\max}, \forall (s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}. \quad (2)$$

A stationary policy maps a state $s \in \mathcal{S}$ to a probability distribution $\pi(\cdot|s)$ over $\mathcal{A}$, which does not depend on time. For a policy π, its value function $V^\pi : \mathcal{S} \to \mathbb{R}$ is defined as the expected accumulated discounted reward by executing the policy π to obtain actions:

$$V^\pi(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(S_t, A_t, S_{t+1}) | S_0 = s_0\right]. \quad (3)$$

The action-value function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ of policy π is defined as

$$Q^\pi(s,a) = \mathbb{E}_{S' \sim \mathsf{P}(\cdot|s,a)}\left[r(s,a,S') + \gamma V^\pi(S')\right]. \quad (4)$$

The goal of optimal control in RL is to find the optimal policy π^* that maximizes the value function for any initial state, i.e., to solve the following problem:

$$V^*(s) = \sup_\pi V^\pi(s), \ \forall s \in \mathcal{S}. \quad (5)$$

We can also define the optimal action-value function as

$$Q^*(s,a) = \sup_\pi Q^\pi(s,a), \ \forall (s,a) \in \mathcal{S} \times \mathcal{A}. \quad (6)$$

Then, the optimal policy π^* is greedy w.r.t. Q^*. The Bellman operator $\mathbf{T}$ is defined as

$$(\mathbf{T}Q)(s,a) = \int_{\mathcal{S}} (r(s,a,s') + \gamma \max_{b \in \mathcal{A}} Q(s',b)) \mathsf{P}(ds'|s,a). \quad (7)$$

It is clear that $\mathbf{T}$ is contraction in the sup norm defined as $\|Q\|_{\sup} = \sup_{(s,a) \in \mathcal{S} \times \mathcal{A}} |Q(s,a)|$, and the optimal action-value function Q^* is the fixed point of $\mathbf{T}$ [22].

2.2 Linear Function Approximation

In many modern RL applications, the state space is usually very large or even continuous. Therefore, classical tabular approach cannot be directly applied due to memory and computational constraint [27]. In this case, the approach of function approximation can be applied, which uses a family of parameterized function to approximate the action-value function. In this paper, we focus on linear function approximation.

Consider a set of N fixed base functions $\phi^{(i)}$: $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$, $i = 1, \ldots, N$. Further consider a family of real-valued functions $\mathcal{Q} = \{Q_\theta : \theta \in \mathbb{R}^N\}$ defined on $\mathcal{S} \times \mathcal{A}$, which consists of linear combinations of $\phi^{(i)}$, $i = 1, \ldots, N$. Specifically,

$$Q_\theta(s,a) = \sum_{i=1}^N \theta(i) \phi^{(i)}_{s,a} = \phi^\top_{s,a} \theta. \quad (8)$$

The goal is to find a Q_θ with a compact representation in θ to approximate the optimal action-value function Q^*.

2.3 Greedy-GQ Algorithm

In this subsection, we introduce the Greedy-GQ algorithm, which was originally proposed in [18] to solve the problem of optimal control in RL under off-policy training.

For the Greedy-GQ algorithm, a fixed behavior policy π_b is used to collect samples. It is assumed that the Markov chain $\{X_t, A_t\}_{t=0}^{\infty}$ induced by the behavior policy π_b and the Markov transition kernel P is uniformly ergodic with the invariant measure denoted by μ.

The main idea of the Greedy-GQ algorithm is to design an objective function, and further to employ a stochastic gradient descent optimization approach together with a weight doubling trick (a two timescale update) [29] to minimize the objective function. Specifically, the goal is to minimize the following mean squared projected Bellman error (MSPBE):

$$J(\theta) \triangleq ||\mathbf{\Pi}\mathbf{T}^{\pi_\theta} Q_\theta - Q_\theta||_\mu. \tag{9}$$

Here $\|Q(\cdot,\cdot)\|_\mu \triangleq \int_{s\in\mathcal{S}, a\in\mathcal{A}} d\mu_{s,a} Q(s,a)$; $\mathbf{T}^\pi$ is the Bellman operator:

$$\mathbf{T}^\pi Q(s,a) \triangleq \mathbb{E}_{S',A'}[r(s,a,S') + \gamma Q(S',A'))], \tag{10}$$

where $S' \sim \mathsf{P}(\cdot|s,a)$, and $A' \sim \pi(\cdot|S')$; $\mathbf{\Pi}$ is a projection operator which projects an action-value function to the function space $\mathcal{Q}$ with respect to $||\cdot||_\mu$, i.e., $\mathbf{\Pi}\hat{Q} = \arg\min_{Q\in\mathcal{Q}} \|Q - \hat{Q}\|_\mu$; and π_θ is a stationary policy, which is a function of θ.

We note that the objective function in (9) is non-convex since the parameter θ is also in the Bellman operator, i.e., π_θ. Moreover, unlike GTD, GTD2 and TDC, the objective function of the Greedy-GQ algorithm is not a quadratic function of θ. Thus, the Greedy-GQ algorithm is not a linear two timescale stochastic approximation algorithm.

Define $\delta_{s,a,s'}(\theta) = r(s,a,s') + \gamma \bar{V}_{s'}(\theta) - \theta^\top \phi_{s,a}$, and $\bar{V}_{s'}(\theta) = \sum_{a'} \pi_\theta(a'|s')\theta^\top \phi_{s',a'}$. In this way, the objective function in (9) can be rewritten equivalently as follows

$$\begin{aligned} J(\theta) =& \mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}]^\top \mathbb{E}_\mu[\phi_{S,A}\phi_{S,A}^\top]^{-1} \\ & \times \mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}], \end{aligned} \tag{11}$$

where $(S,A) \sim \mu$, and $S' \sim \mathsf{P}(\cdot|S,A)$ is the subsequent state.

To compute a gradient to $J(\theta)$, we will need to compute the gradient to $\delta_{S,A,S'}(\theta)$, and thus the gradient to $\bar{V}_{S'}(\theta)$. Suppose $\hat{\phi}_{S'}(\theta)$ is an unbiased estimate of the gradient to $\bar{V}_{S'}(\theta)$ given S', then $\psi_{S,A,S'}(\theta) = \gamma\hat{\phi}_{S'}(\theta) - \phi_{S,A}$ is a gradient of $\delta_{S,A,S'}(\theta)$. Then, the gradient to $J(\theta)/2$ can be computed as follows:

$$\begin{aligned} &\mathbb{E}_\mu[\psi_{S,A,S'}(\theta)\phi_{S,A}^\top]\mathbb{E}_\mu[\phi_{S,A}\phi_{S,A}^\top]^{-1}\mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}] \\ &= -\mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}] + \gamma\mathbb{E}_\mu[\hat{\phi}_{S'}(\theta)\phi_{S,A}^\top]\omega^*(\theta), \end{aligned} \tag{12}$$

where $\omega^*(\theta) = \mathbb{E}_\mu[\phi_{S,A}\phi_{S,A}^\top]^{-1}\mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}]$. To get an unbiased estimate of (12), two independent samples of (S,A,S') are needed, which is not applicable when there is a single sample trajectory. Then, a weight doubling trick [29] was used in [18] to construct the Greedy-GQ algorithm with the following updates (see Algorithm 1 for more details):

$$\theta_{t+1} = \theta_t + \alpha_t(\delta_{t+1}(\theta_t)\phi_t - \gamma(\omega_t^\top\phi_t)\hat{\phi}_{t+1}(\theta_t)), \tag{13}$$

$$\omega_{t+1} = \omega_t + \beta_t(\delta_{t+1}(\theta_t) - \phi_t^\top\omega_t)\phi_t, \tag{14}$$

where $\alpha_t > 0$ and $\beta_t > 0$ are non-increasing step-sizes, $\delta_{t+1}(\theta) \triangleq \delta_{s_t,a_t,s_{t+1}}(\theta)$ and $\phi_t \triangleq \phi_{s_t,a_t}$. For more details of the derivation of the Greedy-GQ algorithm, we refer the readers to [18].

Algorithm 1 Greedy-GQ [18]

Initialization:
$\theta_0, \omega_0, s_0, \phi^{(i)}$, for $i = 1, 2, ..., N$
Method:
$\pi_{\theta_0} \leftarrow \Gamma(\phi^\top\theta_0)$
for $t = 0, 1, 2, ...$ **do**
  Choose a_t according to $\pi_b(\cdot|s_t)$
  Observe s_{t+1} and r_t
  $\bar{V}_{s_{t+1}}(\theta_t) \leftarrow \sum_{a'\in\mathcal{A}} \pi_{\theta_t}(a'|s_{t+1})\theta_t^\top\phi_{s_{t+1},a'}$
  $\delta_{t+1}(\theta_t) \leftarrow r_t + \gamma\bar{V}_{s_{t+1}}(\theta_t) - \theta_t^\top\phi_t$
  $\hat{\phi}_{t+1}(\theta_t) \leftarrow$ gradient of $\bar{V}_{s_{t+1}}(\theta_t)$
  $\theta_{t+1} \leftarrow \theta_t + \alpha_t(\delta_{t+1}(\theta_t)\phi_t - \gamma(\omega_t^\top\phi_t)\hat{\phi}_{t+1}(\theta_t))$
  $\omega_{t+1} \leftarrow \omega_t + \beta_t(\delta_{t+1}(\theta_t) - \phi_t^\top\omega_t)\phi_t$
  Policy improvement: $\pi_{\theta_{t+1}} \leftarrow \Gamma(\phi^\top\theta_{t+1})$
end for

In Algorithm 1, Γ is a policy improvement operator, which maps an action-value function to a policy, e.g., greedy, ϵ-greedy, and softmax and mellowmax [1].

3 FINITE-SAMPLE ANALYSIS FOR GREEDY-GQ

In this section, we will first introduce some technical assumptions, and then present our main results.

We make the following standard assumptions.

Assumption 1 (Problem solvability)**.** *The matrix $C = \mathbb{E}_\mu[\phi_t\phi_t^\top]$ is non-singular.*

Assumption 2 (Bounded feature)**.** $\|\phi_{s,a}\|_2 \leq 1, \forall(s,a) \in \mathcal{S}\times\mathcal{A}$.

Assumption 3 (Geometric uniform ergodicity)**.** *There exists some constants $m > 0$ and $\rho \in (0,1)$ such that*

$$\sup_{s\in\mathcal{S}} d_{TV}(\mathbb{P}(s_t|s_0 = s), \mu) \leq m\rho^t, \tag{15}$$

for any $t > 0$, where d_{TV} is the total-variation distance between the probability measures.

In this paper, we focus on policies that are smooth. Specifically, $\pi_\theta(a|s)$ and $\nabla\pi_\theta(a|s)$ are Lipschitz functions of θ.

Assumption 4 (Policy smoothness)**.** *The policy $\pi_\theta(a|s)$ is k_1-Lipschitz and k_2-smooth, i.e., for any $(s,a)\in\mathcal{S}\times\mathcal{A}$,*

$$\|\nabla\pi_\theta(a|s)\|\leq k_1,\forall\theta, \tag{16}$$

and,

$$\|\nabla\pi_{\theta_1}(a|s)-\nabla\pi_{\theta_2}(a|s)\|\leq k_2\|\theta_1-\theta_2\|,\forall\theta_1,\theta_2. \tag{17}$$

We note that the smaller the k_1 and k_2 are, the smoother the policy is. This family contains many policies as special cases, e.g., softmax and mellowmax [1]. We also note that the greedy policy is not smooth, since it is not differentiable.

To justify the feasibility of Assumption 4 in practice, in the following, we first provide an example of the softmax policy, and show that it is Lipschitz and smooth in θ. Consider the softmax operator, where for any $(a,s)\in\mathcal{A}\times\mathcal{S}$ and $\theta\in\mathbb{R}^N$,

$$\pi_\theta(a|s)=\frac{e^{\sigma\theta^\top\phi_{s,a}}}{\sum_{a'\in\mathcal{A}}e^{\sigma\theta^\top\phi_{s,a'}}}, \tag{18}$$

for some $\sigma>0$.

Lemma 1. *The softmax policy $\pi_\theta(a|s)$ is 2σ-Lipschitz and $8\sigma^2$-smooth, i.e., for any $(s,a)\in\mathcal{S}\times\mathcal{A}$, and for any $\theta_1,\theta_2\in\mathbb{R}^N$,*

$$|\pi_{\theta_1}(a|s)-\pi_{\theta_2}(a|s)|\leq 2\sigma\|\theta_1-\theta_2\|, \tag{19}$$

$$\|\nabla\pi_{\theta_1}(a|s)-\nabla\pi_{\theta_2}(a|s)\|\leq 8\sigma^2\|\theta_1-\theta_2\|. \tag{20}$$

As $\sigma\to\infty$, the softmax policy approximates the greedy policy asymptotically, however its Lipschitz and smoothness constants also go to infinity.

It can be seen from (9) that the objective function of the Greedy-GQ algorithm is non-convex. It may not be possible to guarantee the convergence of the algorithm to the global optimum. Therefore, to measure the convergence rate, we consider the convergence rate of the gradient norm to zero. Furthermore, motivated by the randomized stochastic gradient method in [10], which is designed to analyze non-convex optimization problems, in this paper, we also consider a randomized version of the Greedy-GQ algorithm in Algorithm 1. Specifically, let M be an independent random variable with probability mass function $\mathbb{P}_M$. For steps from 1 to M, call the Greedy-GQ algorithm in Algorithm 1. The final output is then θ_M.

In the following theorem, we provide the convergence rate bound for $\mathbb{E}[\|\nabla J(\theta_M)\|^2]$ when constant step-sizes are used. Specifically, let $M\in\{1,2,...T\}$ and

$$\mathbb{P}(M=k)=\frac{\alpha_k}{\sum_{t=1}^T\alpha_t}. \tag{21}$$

Theorem 1. *Consider the following step-sizes: $\beta=\beta_t=\frac{1}{T^b}$, and $\alpha=\alpha_t=\frac{1}{T^a}$, where $\frac{1}{2}<a\leq 1$ and $0<b\leq a$. Then we have that for $T\geq 1$,*

$$\mathbb{E}[\|\nabla J(\theta_M)\|^2]=\mathcal{O}\left(\frac{1}{T^{1-a}}+\frac{\log T}{T^{\min\{b,a-b\}}}\right). \tag{22}$$

Here we only provide the order of the bound in terms of T. An explicit bound can also be derived, which however is cumbersome and tedious. To understand how different parameters, e.g., L,C,m,ρ, affect the convergence speed, we refer the readers to equation (99) in the appendix.

Although it is not explicitly characterized in (22), we note that as k_1 and k_2 increases, the bound will become looser and thus the algorithm will need more samples to converge. For a more "greedy" target policy with larger k_1 and k_2, it will require more samples to converge. This suggests a practical trade-off between the quality of the obtained policy and the sample complexity.

Theorem 1 characterizes the relationship between the convergence rate and the choice of the step-sizes α_t and β_t. We further optimize over the choice of the step-sizes and obtain the best bound as in the following corollary.

Corollary 1. *If we choose $a=\frac{2}{3}$ and $b=\frac{1}{3}$, then the best rate of the bound in* (22) *is obtained as follows:*

$$\mathbb{E}[\|\nabla J(\theta_M)\|^2]=\mathcal{O}\left(\frac{\log T}{T^{\frac{1}{3}}}\right). \tag{23}$$

For the general non-convex optimization problem with a Lipschitz gradient, the convergence rate of the randomized stochastic gradient method is $\mathcal{O}(T^{-\frac{1}{2}})$ [10]. However, the gradient estimate in that problem is unbiased, and the update is one timescale. In our problem, we have a two timescale update rule. Although the fast timescale updates much faster than the slow timescale, there still exists an estimation error, which we call it "tracking error". Specifically, the tracking error is defined as

$$z_t=w_t-w^*(\theta_t). \tag{24}$$

Moreover, in this paper, we consider the practical scenario where a single sample trajectory with Markovian noise is used. Therefore, for the Greedy-GQ algorithm, there exists bias in the gradient estimate, which justifies the difference in the convergence rate from the one for general non-convex optimization problems [10].

4 PROOF SKETCH

In this section, we provide an outline of the proof, and highlight our major technical contributions. For a complete proof, we refer the readers to the appendix.

The proof can summarized in the following five steps.

1. We first prove that $J(\theta)$ is Lipschitz and smooth.
2. We then decompose the error recursively.
3. We provide a comprehensive characterization of stochastic bias terms and the tracking error in the two timescale updates.
4. We then recursively plug the obtained bound on $\mathbb{E}[\|\nabla J(\theta_M)\|^2]$ back into the analysis, and repeat recursively to obtain the tightest bound.
5. We then optimize the convergence rate over the choice of step-sizes.

In the following, we discuss the proof sketch step by step with more details.

Step 1. We first provide a characterization of the geometric property of the objective function $J(\theta)$. Specifically, we show that if π_θ is Lipschitz and smooth (satisfying Assumption 4), then $J(\theta)$ is also Lipschitz and K-smooth for some $K > 0$, i.e., for any θ_1 and θ_2,

$$\|\nabla J(\theta_1) - \nabla J(\theta_2)\| \leq K||\theta_1 - \theta_2||. \tag{25}$$

Here, larger k_1 and k_2 imply a larger K. As will be seen later in Step 2 and Step 3, a larger K means a looser bound and a higher sample complexity. This theoretical assertion will also be validated in our numerical experiments.

Recall that $J(\theta)$ can be equivalently written as $\mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}]^\top \mathbb{E}_\mu[\phi_{S,A}\phi_{S,A}^\top]^{-1}\mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}]$, which has a quadratic form in $\mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}]$. Therefore, it suffices to show that $\mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_{S,A}]$ is bounded, Lipschitz and smooth, which is clear from its definition and the fact that π_θ is Lipschitz and smooth.

Step 2. Since the object function $J(\theta)$ is Lipschitz and K-smooth, then by Taylor expansion, we have that

$$\begin{aligned} &J(\theta_{t+1}) - J(\theta_t) - \langle \theta_{t+1} - \theta_t, \nabla J(\theta_t)\rangle \\ &\leq \frac{K}{2}\|\theta_{t+1} - \theta_t\|^2. \end{aligned} \tag{26}$$

Denote by $G_{t+1}(\theta,\omega) = (\delta_{t+1}(\theta)\phi_t - \gamma(\omega^\top\phi_t)\hat{\phi}_{t+1}(\theta))$. Then, the difference between θ_t and θ_{t+1} is $\alpha_t G_{t+1}(\theta_t,\omega_t)$. The inequality (26) can be further written as

$$\begin{aligned} &J(\theta_{t+1}) - J(\theta_t) - \alpha_t\langle G_{t+1}(\theta_t,\omega_t), \nabla J(\theta_t)\rangle \\ &\leq \frac{K\alpha_t^2}{2}\|G_{t+1}(\theta_t,\omega_t)\|^2. \end{aligned} \tag{27}$$

Note that $G_{t+1}(\theta_t,\omega_t)$ is the stochastic gradient used in the Greedy-GQ algorithm. Due to the two timescale update and the Markovian noise, the stochastic gradient is biased. For a finite-sample analysis, we will then need to characterize the stochastic bias in the gradient estimate $G_{t+1}(\theta_t,\omega_t)$ explicitly.

We first consider the difference between the true gradient $\nabla J(\theta_t)$ and the gradient estimate $G_{t+1}(\theta_t,\omega_t)$ used in the Greedy-GQ algorithm, which is denoted by $\mathbf{\Delta}_t = -2G_{t+1}(\theta_t,\omega_t) - \nabla J(\theta_t)$. Plug this in the inequality (27), and we obtain that

$$\begin{aligned} &J(\theta_{t+1}) - J(\theta_t) + \frac{\alpha_t}{2}\langle(\mathbf{\Delta}_t + \nabla J(\theta_t)), \nabla J(\theta_t)\rangle \\ &= J(\theta_{t+1}) - J(\theta_t) + \frac{\alpha_t}{2}\|\nabla J(\theta_t)\|^2 \\ &\quad + \alpha_t\left\langle\frac{1}{2}\mathbf{\Delta}_t, \nabla J(\theta_t)\right\rangle \\ &\leq \alpha_t^2\frac{K}{2}\|G_{t+1}(\theta_t,\omega_t)\|^2. \end{aligned} \tag{28}$$

Recall the definition of the random variable M in (21). Applying (28) recursively, we have that

$$\begin{aligned} &\mathbb{E}[\|\nabla J(\theta_M)\|^2] \\ &\leq \frac{1}{\sum_{t=0}^T \alpha_t}\Bigg((J(\theta_0) - J(\theta_{T+1})) \\ &\qquad + \frac{K}{2}\sum_{t=0}^T \alpha_t^2\mathbb{E}[\|G_{t+1}(\theta_t,\omega_t)\|^2] \\ &\qquad - \sum_{t=0}^T \frac{\alpha_t}{2}\langle\mathbf{\Delta}_t, \nabla J(\theta_t)\rangle\Bigg). \end{aligned} \tag{29}$$

From (29), it can be seen that to understand the convergence rate of $\mathbb{E}[\|\nabla J(\theta_M)\|^2]$, we need to bound the three terms on the right hand side of (29). The first and second terms are straightforward to bound since $J(\theta)$ is non-negative for any θ, and $\|G_{t+1}\|$ is uniformly bounded by some constant.

For the third term $\langle\mathbf{\Delta}_t, \nabla J(\theta_t)\rangle$, it can be further decomposed into the following two parts

$$\begin{aligned} &\big\langle\nabla J(\theta_t), -2G_{t+1}(\theta_t,\omega_t) + 2G_{t+1}(\theta_t,\omega^*(\theta_t))\big\rangle \\ &- \big\langle\nabla J(\theta_t), \nabla J(\theta_t) + 2G_{t+1}(\theta_t,\omega^*(\theta_t))\big\rangle, \end{aligned} \tag{30}$$

where the first part is corresponding to the tracking error, and the second part is corresponding to the stochastic bias caused by the Markovian noise.

Step 3. We then provide bounds for each term in (29) and (30). For the first and second terms in (29), it is straightforward to develop their upper bounds. For the first term in (30), it can be upper bounded by exploiting the Lipschitz property of $G_{t+1}(\theta,\omega)$ in ω. Specifically,

$$\begin{aligned}&\left\langle \nabla J(\theta_t), -2G_{t+1}(\theta_t,\omega_t) + 2G_{t+1}(\theta_t,\omega^*(\theta_t))\right\rangle\\ &\leq \xi_1 \|\nabla J(\theta_t)\|\|\omega_t - \omega^*(\theta_t)\|,\end{aligned} \tag{31}$$

for some $\xi_1 > 0$. Thus, it suffices to bound the tracking error $\|\omega_t - \omega^*(\theta_t)\|$. The bound on the tracking error is difficult due to the complicated coupling between the parameter ω_t, θ_t and the sample trajectory. We decouple such the dependence between ω_t, θ_t and the samples by looking τ steps back, where τ is the mixing time of the MDP. By the geometric uniform ergodicity, conditioning on $\omega_{t-\tau}$ and $\theta_{t-\tau}$, the distribution of (s_t, a_t) is close to the stationary distribution μ. Thus, the expectation of the tracking error can be bounded.

We then bound the second term in (30). We know that for any fixed θ, $\mathbb{E}_\mu[\nabla J(\theta) + 2G_{t+1}(\theta,\omega^*(\theta))] = 0$. However, θ_t and S_t, A_t, S_{t+1} are not independent. Similarly, we exploit the geometric uniform ergodicity of the MDP. For simplicity, we denote by

$$\zeta(\theta_t, O_t) = \left\langle \nabla J(\theta_t), \nabla J(\theta_t) + 2G_{t+1}(\theta_t,\omega^*(\theta_t))\right\rangle, \tag{32}$$

where $O_t = \{S_t, A_t, S_{t+1}, r_t\}$. We can show that $\zeta(\theta, O_t)$ is Lipschitz in θ. Thus, if we look τ step back, then

$$|\zeta(\theta_t, O_t) - \zeta(\theta_{t-\tau}, O_t)| \leq c_\zeta \|\theta_t - \theta_{t-\tau}\|, \tag{33}$$

for some $c_\zeta > 0$. Therefore,

$$\zeta(\theta_t, O_t) \leq \zeta(\theta_{t-\tau}, O_t) + c_\zeta \|\theta_t - \theta_{t-\tau}\|. \tag{34}$$

Since we are using small step-sizes, then $\|\theta_t - \theta_{t-\tau}\|$ should be small. In other words, the difference between $\zeta(\theta_t, O_t)$ and $\zeta(\theta_t, O_t)$ is small. By the geometric uniform ergodicity, for any $\theta_{t-\tau}$, the distribution of O_t is close to the stationary distribution μ. Thus, even $\theta_{t-\tau}$ and O_t are not independent, we can still upper bound $\mathbb{E}[\zeta(\theta_{t-\tau}, O_t)]$. In this way, we decouple the dependence between θ_t and O_t, and we can obtain the bound on the gradient bias.

Step 4. After Step 3, we can obtain the following bound on $\mathbb{E}[\|\nabla J(\theta_M)\|^2]$:

$$\mathbb{E}[\|\nabla J(\theta_M)\|^2] = \mathcal{O}\left(\frac{1}{T^{1-a}} + \frac{\sqrt{\log T}}{T^{\frac{1}{2}\min\{b,a-b\}}}\right). \tag{35}$$

This bound is obtained by upper bounding $\|\nabla J(\theta_t)\|$ on the right hand side of (29) using a constant. Obviously, $\mathbb{E}[\|\nabla J(\theta_M)\|^2] \to 0$ as $T \to \infty$, and thus using a constant to upper bound $\nabla J(\theta_t)$ is not tight.

In this step, we recursively use the obtained bound to further tighten the bound on $\mathbb{E}[\|\nabla J(\theta_M)\|^2]$. Specifically, we plug (35) back into (31) in Step 3. If $1 - a > \min\{b, a - b\}$, then the second term on the right hand side of (35) dominates. Plugging (35) back into (31) will further tighten the bound to the following one:

$$\mathbb{E}[\|\nabla J(\theta_M)\|^2] = \mathcal{O}\left(\frac{1}{T^{1-a}} + \frac{\log^{\frac{3}{4}} T}{T^{\frac{3}{4}\min\{b,a-b\}}}\right). \tag{36}$$

Repeat this procedure, we can then obtain the following bound:

$$\mathbb{E}[\|\nabla J(\theta_M)\|^2] = \mathcal{O}\left(\frac{1}{T^{1-a}} + \frac{\log T}{T^{\min\{b,a-b\}}}\right). \tag{37}$$

If $1 - a \leq \frac{1}{2}\min\{b, a - b\}$, then the first term in (35) dominates. Therefore, the above recursive refinement will not improve the convergence rate. If $\frac{1}{2}\min\{b, a - b\} \leq 1 - a \leq \min\{b, a - b\}$, we can apply our recursive bounding trick finite times until the first term $\mathcal{O}\left(\frac{1}{T^{1-a}}\right)$ in (35) dominates. Combining the analyses for the three cases, the overall convergence rate bound can be obtained, which is as in (37).

Step 5. Given the convergence rate bound in (37), in this step, we optimize over the choice of the step-sizes to obtain the fastest convergence rate. Recall that $\frac{1}{2} < a \leq 1$ and $0 < b \leq a$. Then, it can be derived that when $a = \frac{2}{3}$ and $b = \frac{1}{3}$, the best convergence rate that is achievable in (37) is $\mathcal{O}\left(\frac{\log T}{T^{\frac{1}{3}}}\right)$.

5 NUMERICAL EXPERIMENTS

In this section, we present our numerical experiments. Specifically, we investigate how the Lipschitz and smoothness constants affect the convergence of the Greedy-GQ algorithm. We use the the softmax operator as an example. Recall that in Lemma 1, the Lipschitz and smoothness constants of the softmax operator is an increasing function of σ in (18).

As has been observed in our finite-sample analysis, the upper bound on the gradient norm increases with K, and thus increases with σ. This suggests a higher sample complexity as the target policy becomes more "greedy". We will numerically validate this observation by simulating the Greedy-GQ algorithm for different values of σ in (18).

We consider a simple example: $\mathcal{S} = \{1, 2, 3, 4\}$ and $\mathcal{A} = \{1, 2\}$. For the first MDP we consider, taking any action at any state will have the same probability to transit to any

state, i.e. $\mathbb{P}(s'|s,a) = \frac{1}{4}$ for any (s,a,s'). Five different values of σ are considered: $\sigma = 1, 2, 3, 15, 20$.

We randomly generate two base functions. We initialize $s_0 = 2$, $\theta_0 = (1,2)^\top$ and $\omega_0 = (0.1, 0.1)^\top$. At each iteration, we choose $A_t \sim \pi_b$, update θ_{t+1} and ω_{t+1} according to Algorithm 1, and compute $\|\nabla J(\theta_t)\|^2$. As for T, we consider $T = 1000$.

For the same state and action spaces, we vary the behavior policy and Markov transition kernel, and repeat our experiment for three more times, the more specific settings are followed:
MDP1: $\pi_b(a|s) = 0.5$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ and $\mathbb{P}(s'|s,a) = 0.25$ for all $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$;
MDP2: $\pi_b(a=1|s) = 0.4$, $\pi_b(a=2|s) = 0.6$ for all $s \in \mathcal{S}$ and $\mathbb{P}(s'|s,a) = 0.25$ for all $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$;
MDP3: $\pi_b(a|s) = 0.5$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, $\mathbb{P}(s|s,a=1) = 1$ for all $s \in \mathcal{S}$ and $\mathbb{P}(s'|s,a=2) = 0.25$ for all $(s,s') \in \mathcal{S} \times \mathcal{S}$;
MDP4: $\pi_b(a=1|s) = 0.3$, $\pi_b(a=2|s) = 0.7$ for all $s \in \mathcal{S}$ and $\mathbb{P}(s'|s,a) = 0.25$ for all $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$.

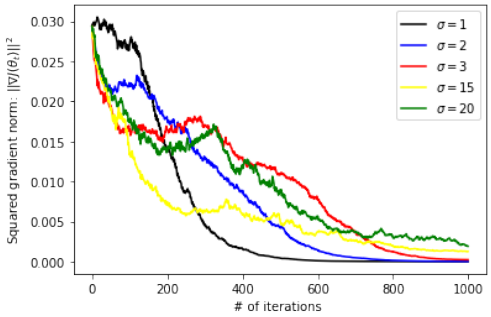
For all the four MDPs, we set $r(s,a,s') = 1$ for all $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$.

We plot the gradient norm as a function of the number of iterations in Fig. 1.

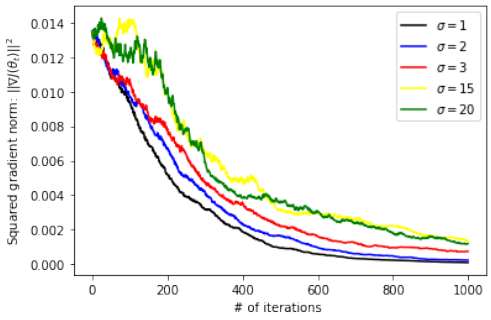
It can be seen from Fig. 1, as σ increases, the convergence of the Greedy-GQ algorithm is getting slower. This observation matches with our theoretical bound that the Greedy-GQ algorithm has a higher sample complexity if the targeted policy is less smoother.

6 CONCLUSION

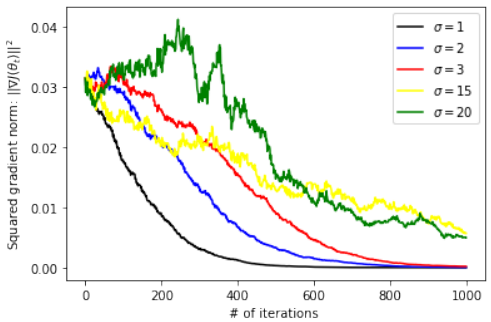
In this paper, we developed the first finite-sample analysis for the Greedy-GQ algorithm with linear function approximation under Markovian noise. Our analysis is from a novel optimization perspective to solve RL problems. We comprehensively characterized the stochastic bias in the gradient estimate and designed a novel technique which recursively applies the obtained bound back into the bias analysis to tighten the convergence rate bound. We characterized the convergence rate of the Greedy-GQ algorithm, and provided a general guide for choosing step-sizes in practice. The convergence rate obtained by our analysis is $\mathcal{O}\left(\frac{\log T}{T^{\frac{1}{3}}}\right)$, and is close to the convergence rate $\mathcal{O}\left(\frac{1}{T^{\frac{1}{2}}}\right)$ for general non-convex optimization problems with unbiased gradient estimate. Such a different is mainly due to the Markovian noise and the tracking error in the two timescale updates. The techniques developed in this paper may be of independent interest for a wide range of reinforcement learning problems with non-convex objective function and Markovian noise.



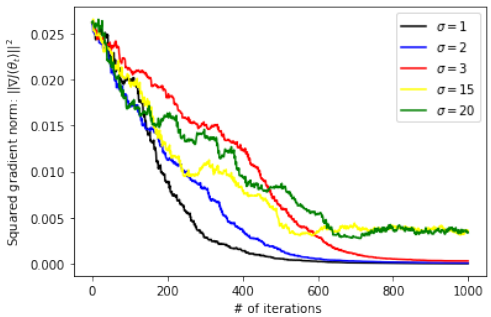
(a) MDP 1



(b) MDP 2



(c) MDP 3



(d) MDP 4

Figure 1: Comparison among different σ for the Greedy-GQ algorithm with softmax operator.

In this paper, we provided the finite-sample analysis and the convergence rate for the case with constant step-sizes. The convergence rate for the case with diminishing step-sizes can be derived similarly. One interesting future direction is to investigate the Greedy-GQ algorithm with the greedy policy. Specifically,

$$\pi_\theta(a|s) = 1 \text{ if } a = \arg\max_{a' \in \mathcal{A}} \phi_{s,a}^\top \theta.$$

Due to this max operator, the objective function $J(\theta)$ becomes non-differentiable and non-smooth. To the best of the author's knowledge, there does not exist a general methodology to analyze non-convex non-differentiable optimization problems. One possible solution is to explore the special geometry of the objective function, i.e., $J(\theta)$ is a piece-wise quadratic function of θ. It is also of further interest to investigate the Greedy-GQ algorithm with general function approximation, e.g., neural network.

References

[1] K. Asadi and M. L. Littman. An alternative softmax operator for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 243–252. JMLR. org, 2017.

[2] L. Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

[3] J. Bhandari, D. Russo, and R. Singal. A finite time analysis of temporal difference learning with linear function approximation. *arXiv preprint arXiv:1806.02450*, 2018.

[4] V. S. Borkar. *Stochastic approximation: a dynamical systems viewpoint*, volume 48. Springer, 2009.

[5] V. S. Borkar and S. Pattathil. Concentration bounds for two time scale stochastic approximation. In *Proc. Annu. Allerton Conf. Communication, Control and Computing*, pages 504–511. IEEE, 2018.

[6] Q. Cai, Z. Yang, J. D. Lee, and Z. Wang. Neural temporal-difference learning converges to global optima. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 11312–11322, 2019.

[7] Z. Chen, S. Zhang, T. T. Doan, S. T. Maguluri, and J.-P. Clarke. Performance of Q-learning with linear function approximation: Stability and finite-time analysis. *arXiv preprint arXiv:1905.11425*, 2019.

[8] G. Dalal, B. Szörényi, G. Thoppe, and S. Mannor. Finite sample analysis of two-timescale stochastic approximation with applications to reinforcement learning. *Proceedings of Machine Learning Research*, 75:1–35, 2018.

[9] G. Dalal, B. Szrnyi, G. Thoppe, and S. Mannor. Finite sample analyses for TD(0) with function approximation. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, pages 6144–6160, 2018.

[10] S. Ghadimi and G. Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[11] G. J. Gordon. Chattering in SARSA (λ)-a CMU learning lab internal report. 1996.

[12] H. Gupta, R. Srikant, and L. Ying. Finite-time performance bounds and adaptive learning rate selection for two time-scale reinforcement learning. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 4706–4715, 2019.

[13] P. Karmakar and S. Bhatnagar. Two time-scale stochastic approximation with controlled Markov noise and off-policy temporal-difference learning. *Mathematics of Operations Research*, 43(1):130–151, 2018.

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

[15] C. Lakshminarayanan and C. Szepesvari. Linear stochastic approximation: How far does constant step-size and iterate averaging go? In *Proc. International Conference on Artifical Intelligence and Statistics (AISTATS)*, 2018.

[16] B. Liu, J. Liu, M. Ghavamzadeh, S. Mahadevan, and M. Petrik. Finite-sample analysis of proximal gradient td algorithms. In *Proc. International Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 504–513. Citeseer, 2015.

[17] H. R. Maei. Gradient temporal-difference learning algorithms. *Thesis, University of Alberta*, 2011.

[18] H. R. Maei, C. Szepesvári, S. Bhatnagar, and R. S. Sutton. Toward off-policy learning control with function approximation. In *Proc. International Conference on Machine Learning (ICML)*, 2010.

[19] F. S. Melo, S. P. Meyn, and M. I. Ribeiro. An analysis of reinforcement learning with function approximation. In *Proc. International Conference on Machine Learning (ICML)*, pages 664–671. ACM, 2008.

[20] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proc. International Conference on Machine Learning (ICML)*, pages 1928–1937, 2016.

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, and G. Ostrovski. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

[22] D. P and Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

[23] T. J. Perkins and D. Precup. A convergent form of approximate policy iteration. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 1627–1634, 2003.

[24] G. A. Rummery and M. Niranjan. Online Q-learning using connectionist systems. *Technical Report, Cambridge University Engineering Department*, Sept. 1994.

[25] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, and M. Lanctot. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[26] R. Srikant and L. Ying. Finite-time error bounds for linear stochastic approximation and TD learning. In *Proc. Annual Conference on Learning Theory (CoLT)*, 2019.

[27] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction, Second Edition*. The MIT Press, Cambridge, Massachusetts, 2018.

[28] R. S. Sutton, H. R. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvári, and E. Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proc. International Conference on Machine Learning (ICML)*, pages 993–1000, 2009.

[29] R. S. Sutton, H. R. Maei, and C. Szepesvári. A convergent $O(n)$ temporal-difference algorithm for off-policy learning with linear function approximation. In *Advances in neural information processing systems*, pages 1609–1616, 2009.

[30] J. N. Tsitsiklis and B. Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, May 1997.

[31] Y. Wang, W. Chen, Y. Liu, Z.-M. Ma, and T.-Y. Liu. Finite sample analysis of the gtd policy evaluation algorithms in markov setting. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 5504–5513, 2017.

[32] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[33] T. Xu, S. Zou, and Y. Liang. Two time-scale off-policy TD learning: Non-asymptotic analysis over Markovian samples. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 10633–10643, 2019.

[34] H. Yu. On convergence of some gradient-based temporal-differences algorithms for off-policy learning. *arXiv preprint arXiv:1712.09652*, 2017.

[35] S. Zou, T. Xu, and Y. Liang. Finite-sample analysis for SARSA with linear function approximation. In *Advances in Neural Information Processing Systems*, pages 8665–8675, 2019.