Online Robust Reinforcement Learning with Model Uncertainty

Anonymous Author(s)
Affiliation
Address
email

Abstract

Robust reinforcement learning (RL) is to find a policy that optimizes the worst-case performance over an uncertainty set of MDPs. In this paper, we focus on *model-free* robust RL, where the uncertainty set is defined to be centering at a misspecified MDP that generates a single sample trajectory sequentially, and is assumed to be *unknown*. We develop a sample-based approach to estimate the unknown uncertainty set, and design robust Q-learning algorithm (tabular case) and robust TDC algorithm (function approximation setting), which can be implemented in an online and incremental fashion. For the robust Q-learning algorithm, we prove that it converges to the optimal robust Q function, and for the robust TDC algorithm, we prove that it converges asymptotically to some stationary points. Unlike the results in [48], our algorithms do not need any additional conditions on the discount factor to guarantee the convergence. We further characterize the finite-time error bounds of the two algorithms, and show that both the robust Q-learning and robust TDC algorithms converge as fast as their vanilla counterparts (within a constant factor). Our numerical experiments further demonstrate the robustness of our algorithms. Our approach can be readily extended to robustify many other algorithms, e.g., TD, SARSA, and other GTD algorithms.

1 Introduction

Existing studies on Markov decision process (MDP) and reinforcement learning (RL) [52] mostly rely on the crucial assumption that the environment on which a learned policy will be deployed is the same one that was used to generate the policy, which is often violated in practice – e.g., the simulator may be different from the true environment, and the MDP may evolve over time. Due to such model deviation, the actual performance of the learned policy can significantly degrade. To address this problem, the framework of robust MDP was formulated in [5, 41, 25], where the transition kernel of the MDP is not fixed and lies in an uncertainty set, and the goal is to learn a policy that performs well under the worst-case MDP in the uncertainty set. In [5, 41, 25], it was assumed that the uncertainty set is known beforehand, i.e., model-based approach, and dynamic programming can be used to find the optimal robust policy.

The model-based approach, however, requires a model of the uncertainty set known beforehand, and needs a large memory to store the model when the state and action spaces are large, which make it less applicable for many practical scenarios. This motivates the study in this paper, *model-free* robust RL with model uncertainty, which is to learn a robust policy using a single sample trajectory from a misspecified MDP, e.g., a simulator and a similar environment in which samples are easier to collect than in the target environment where the policy is going to be deployed. The major challenge lies in that the transition kernel of the misspecified MDP is not given beforehand, and thus, the uncertainty set and the optimal robust policy need to be learned simultaneously using sequentially observed data

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

from the misspecified MDP. Moreover, robust RL learns the value function of the worst-case MDP in the uncertainty set which is different from the misspecified MDP that generates samples. This is similar to the off-policy learning, which we refer to as the "off-transition-kernel" setting. Therefore, the learning may be unstable and could diverge especially when function approximation is used [6].

In this paper, we develop a model-free approach for robust RL with model uncertainty. Our major contributions in this paper are summarized as follows.

- Motivated by empirical studies of adversarial training in RL [23, 27, 32, 43, 38] and the R-contamination model in robust detection (called ϵ-contamination model in [24]), we design the uncertainty set using the R-contamination model (see (4) for the details). We then develop an approach to estimate the unknown uncertainty set using only the current sample, which does not incur any additional memory cost. Unlike the approach in [48], where the uncertainty set is relaxed to one not depending on the misspecified MDP that generates samples so that an online algorithm can be constructed, our approach does not need to relax the uncertainty set.
- We develop a robust Q-learning algorithm for the tabular case, which can be implemented in an online and incremental fashion, and has the same memory cost as the vanilla Q-learning algorithm. We show that our robust Q-learning algorithm converges asymptotically, and further characterize its finite-time error bound. Unlike the results in [48] where a stringent condition on the discount factor (which is due to the relaxation of the uncertainty set, and prevents the use of a discount factor close to 1 in practice) is needed to guarantee the convergence, our algorithm converges without the need of such condition. Furthermore, our robust Q-learning algorithm converges as fast as the vanilla Q-learning algorithm [30] (within a constant factor), while being robust to model uncertainty.
- We generalize our approach to the case with function approximation (for large state/action space). We investigate the robust policy evaluation problem, i.e., evaluate a given policy under the worst-case MDP in the uncertainty set. As mentioned before, the robust RL problem is essentially "off-transition-kernel", and therefore non-robust methods with function approximation may diverge [6] (also see our experiments). We develop a novel extension of the gradient TD (GTD) method [37, 36, 54] to robust RL. Our approach introduces a novel smoothed robust Bellman operator to construct the smoothed mean-squared projected robust Bellman error (MSPRBE). Using our uncertainty set design and online sample-based estimation, we develop a two time-scale robust TDC algorithm. We further characterize its convergence and finite-time error bound.
- We conduct numerical experiments to validate the robustness of our approach. In our experiments, our robust Q-learning algorithm achieves a much higher reward than the vanilla Q-learning algorithm when being trained on a misspecified MDP; and our robust TDC algorithm converges much faster than the vanilla TDC algorithm, and the vanilla TDC algorithm may even diverge.

1.1 Related Work

Model-Based Robust MDP. The framework of robust MDP was investigated in [25, 42, 5, 49, 62], where the transition kernel is assumed to be in some uncertainty set, and the problem can be solved by dynamic programming. This approach was further extended to the case with function approximation in [55]. However, these studies are model-based, which assume beforehand knowledge of the uncertainty set. In this paper, we investigate the model-free setting, where the uncertainty set is a set of MDPs centered around some unknown Markov transition kernel from which a single sample trajectory can be sequentially observed.

Adversarial Robust RL. It was shown in [25] that the robust MDP problem is equivalent to a zero-sum game between the agent and the nature. Motivated by this fact, the adversarial training approach, where an adversary perturbs the state transition, was studied in [56, 44, 1, 22, 47, 4, 40]. This method relies on a simulator, where the state transition can be modified in an arbitrary way. Another approach is to modify the current state through adversarial samples, which is more heuristic, e.g., [23, 27, 32, 43, 38]. Despite the empirical success of these approaches, theoretical performance guarantees, e.g., convergence to the optimal robust policy and convergence rate, are yet to be established. The main difference lies in that during the training, our approach does not need to manipulate the state transition of the MDP. More importantly, we develop the asymptotic convergence to the optimal robust policy and further characterize the finite-time error bound. In [31], the scenario where some unknown parts of the state space can have arbitrary transitions while other parts are purely stochastic was studied. Adaptive algorithm to adversarial behavior was designed, and its regret

bound is shown to be similar to the purely stochastic case. In [67], the robust adversarial RL problem for the special linear quadratic case was investigated.

Model-free Robust RL. In [48] model-free RL with model uncertainty was studied, where in order to construct an algorithm that can be implemented in an online and incremental fashion, the uncertainty set was firstly relaxed by dropping the dependency on the misspecified MDP that generates the samples (centroid of the uncertainty set). Such a relaxation is pessimistic since the relaxed uncertainty set is not centered at the misspecified MDP anymore (which is usually similar to the target MDP), making the robustness to the relaxed uncertainty set not well-justified. Such a relaxation will further incur a stringent condition on the discounted factor to guarantee the convergence, which prevents the use of a discount factor close to 1 in practice. Moreover, only asymptotic convergence was established in [48]. In this paper, we do not relax the uncertainty set, and instead propose an online approach to estimate it. Our algorithms converge without the need of the condition on the discount factor. We also provides finite-time error bounds for our algorithms. The multi-agent RL robust to reward uncertainty was investigated in [68], where the reward uncertainty set is known, but the transition kernel is fixed.

Finite-time Error Bound for RL Algorithms. For the tubular case, Q-learning has been studied intensively, e.g., in [20, 7, 46, 30, 58, 29]. TD with function approximation were studied in [18, 9, 50, 13, 51]. Q-learning and SARSA with linear function approximation were investigated in [69, 15]. The finite-time error bounds for the gradient TD algorithms [37, 53, 37] were further developed recently in [17, 33, 21, 65, 16, 26, 34, 60, 35, 19]. There are also finite-time error bounds on the policy gradient methods and actor critic methods, e.g., [59, 66, 28, 45, 63, 14, 8, 2, 39]. We note that these studies are for the non-robust RL algorithms, and in this paper, we design robust RL algorithms, and characterize their finite-time error bounds.

2 Preliminaries

2.1 Markov Decision Process

An MDP can be characterized by a tuple $(\mathcal{S}, \mathcal{A}, \mathsf{P}, c, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $\mathsf{P} = \left\{p_s^a \in \Delta_{|\mathcal{S}|}, a \in \mathcal{A}, s \in \mathcal{S}\right\}$ is the transition kernel[1], c is the cost function, and $\gamma \in [0, 1)$ is the discount factor. Specifically, p_s^a denotes the distribution of the next state if taking action a at state s. Let $p_s^a = \{p_{s,s'}^a\}_{s' \in \mathcal{S}}$, where $p_{s,s'}^a$ denotes the probability that the environment transits to state s' if taking action a at state s. The cost of taking action a at state s is given by $c(s, a)$. A stationary policy π is a mapping from $\mathcal{S}$ to a distribution over $\mathcal{A}$. At each time t, an agent takes an action $A_t \in \mathcal{A}$ at state $S_t \in \mathcal{S}$. The environment then transits to the next state S_{t+1} with probability $p_{S_t,S_{t+1}}^{A_t}$, and the agent receives cost given by $c(S_t, A_t)$. The value function of a policy π starting from any initial state $s \in \mathcal{S}$ is defined as the expected accumulated discounted cost by following π: $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t c(S_t, A_t) | S_0 = s, \pi\right]$, and the goal is to find the policy π that minimizes the above value function for any initial state $s \in \mathcal{S}$.

2.2 Robust Markov Decision Process

In the robust case, the transition kernel is not fixed and lies in some uncertainty set. Denote the transition kernel at time t by P_t, and let $\kappa = (\mathsf{P}_0, \mathsf{P}_1, ...)$, where $\mathsf{P}_t \in \mathbf{P}, \forall t \geq 0$, and $\mathbf{P}$ is the uncertainty set of the transition kernel. The sequence κ can be viewed as the policy of the nature, and is adversarially chosen by the nature [5, 41, 25]. Define the robust value function of a policy π as the worst-case expected accumulated discounted cost following a fixed policy π over all transition kernels in the uncertainty set:

$$V^{\pi}(s) = \max_{\kappa} \mathbb{E}_{\kappa}\left[\sum_{t=0}^{\infty} \gamma^t c(S_t, A_t) | S_0 = s, \pi\right], \tag{1}$$

where $\mathbb{E}_{\kappa}$ denotes the expectation when the state transits according to κ. Similarly, define the robust action-value function for a policy π: $Q^{\pi}(s, a) = \max_{\kappa} \mathbb{E}_{\kappa}\left[\sum_{t=0}^{\infty} \gamma^t c(S_t, A_t) | S_0 = s, A_0 = a, \pi\right]$. The goal of robust RL is to find the optimal robust policy π^* that minimizes the worst-case accumulated discounted cost:

$$\pi^* = \arg\min_{\pi} V^{\pi}(s), \forall s \in \mathcal{S}. \tag{2}$$

[1] Δ_n denotes the probability simplex in $\mathbb{R}^n$.

We also denote V^{π^*} and Q^{π^*} by V^* and Q^*, respectively, and $V^*(s) = \min_{a\in\mathcal{A}} Q^*(s,a)$.

Note that a transition kernel is a collection of conditional distributions. Therefore, the uncertainty set $\mathbf{P}$ of the transition kernel can be equivalently written as a collection of $\mathcal{P}_s^a$ for all $s \in \mathcal{S}, a \in \mathcal{A}$, where $\mathcal{P}_s^a$ is a set of conditional distributions p_s^a over the state space $\mathcal{S}$. Denote by $\sigma_{\mathcal{P}}(v) \triangleq \max_{p\in\mathcal{P}}(p^\top v)$ the support function of vector v over a set of probability distributions $\mathcal{P}$. For robust MDP, the following robust analogue of the Bellman recursion was provided in [41, 25].

Theorem 1. *[41] The following perfect duality condition holds for all $s \in \mathcal{S}$:*

$$\min_{\pi}\max_{\kappa}\mathbb{E}_\kappa\left[\sum_{t=0}^{\infty}\gamma^t c(S_t,A_t)\big|\pi, S_0=s\right] = \max_{\kappa}\min_{\pi}\mathbb{E}_\kappa\left[\sum_{t=0}^{\infty}\gamma^t c(S_t,A_t)\big|\pi, S_0=s\right]. \quad (3)$$

The optimal robust value function V^* satisfies $V^*(s) = \min_{a\in\mathcal{A}}(c(s,a) + \gamma\sigma_{\mathcal{P}_s^a}(V^*))$, and the optimal robust action-value function Q^* satisfies $Q^*(s,a) = c(s,a) + \gamma\sigma_{\mathcal{P}_s^a}(V^*)$.

Define the robust Bellman operator $\mathbf{T}$ by $\mathbf{T}Q(s,a) = c(s,a) + \gamma\sigma_{\mathcal{P}_s^a}(\min_{a\in\mathcal{A}} Q(s,a))$. It was shown in [41, 25] that $\mathbf{T}$ is a contraction and its fixed point is the optimal robust Q^*. When the uncertainty set is known, so that $\sigma_{\mathcal{P}_s^a}$ can be computed exactly, V^* and Q^* can be solved by dynamic programming [25, 41].

3 R-Contamination Model For Uncertainty Set Construction

In this section, we construct the uncertainty set using the R-contamination model.

Let $\mathsf{P} = \{p_s^a, s \in \mathcal{S}, a \in \mathcal{A}\}$ be the centroid of the uncertainty set, i.e., the transition kernel that generates the sample trajectory, and P is *unknown*. For example, P can be the simulator at hand, which may not be exactly accurate; and P can be the transition kernel of environment 1, from which we can take samples to learn a policy that will be deployed in a similar environment 2. The goal is to learn a policy using samples from P that performs well when applied to a perturbed MDP from P.

Motivated by empirical studies of adversarial training in RL [23, 27, 32, 43, 38] and the R-contamination model in robust detection [24], we use the R-contaminated model to define the uncertainty set:

$$\mathcal{P}_s^a = \left\{(1-R)p_s^a + Rq | q \in \Delta_{|\mathcal{S}|}\right\}, s \in \mathcal{S}, a \in \mathcal{A}, \text{ for some } 0 \le R \le 1. \quad (4)$$

Here, p_s^a is the centroid of the uncertainty set $\mathcal{P}_s^a$ at (s,a), which is *unknown*, and R is the design parameter of the uncertainty set, which measures the size of the uncertainty set, and is assumed to be known in the algorithm. We then let $\mathbf{P} = \bigotimes_{s\in\mathcal{S},a\in\mathcal{A}} \mathcal{P}_s^a$.

4 Tabular Case: Robust Q-Learning

In this section, we focus on the tabular case with finite state and action spaces. We focus on the asynchronous setting where a single sample trajectory is available with Markovian noise. We will develop an efficient approach to estimate the unknown uncertainty set $\mathbf{P}$, and further the support function $\sigma_{\mathcal{P}_s^a}(\cdot)$, and then design our robust Q-learning algorithm.

We propose an efficient and data-driven approach to estimate the unknown p_s^a and thus the unknown uncertainty set $\mathcal{P}_s^a$ for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$. Specifically, denote the sample at t-th time step by $O_t = (s_t, a_t, s_{t+1})$. We then use O_t to obtain the maximum likelihood estimate (MLE) $\hat{p}_t \triangleq \mathbb{1}_{s_{t+1}}$ of the transition kernel $p_{s_t}^{a_t}$, where $\mathbb{1}_{s_{t+1}}$ is a probability distribution taking probability 1 at s_{t+1} and 0 at other states. This is an unbiased estimate of the transition kernel $p_{s_t}^{a_t}$ conditioning on $S_t = s_t$ and $A_t = a_t$. We then design a sample-based estimate

$$\hat{\mathcal{P}}_t \triangleq \left\{(1-R)\hat{p}_t + Rq | q \in \Delta_{|\mathcal{S}|}\right\} \quad (5)$$

of the uncertainty set $\mathcal{P}_{s_t}^{a_t}$.

Using the sample-based uncertainty set $\hat{\mathcal{P}}_t$, we construct the following robust Q-learning algorithm in Algorithm 1. For any t, $\sigma_{\hat{\mathcal{P}}_t}(V_t)$ can be easily computed: $\sigma_{\hat{\mathcal{P}}_t}(V_t) = R\max_{s\in\mathcal{S}} V_t(s) + (1-R)V_t(s_{t+1})$. Hence the update in Algorithm 1 (line 5) can be explicitly written as

$$Q_{t+1}(s_t,a_t) \leftarrow (1-\alpha_t)Q_t(s_t,a_t) + \alpha_t(c_t + \gamma R\max_{s\in\mathcal{S}} V_t(s) + \gamma(1-R)V_t(s_{t+1})). \quad (6)$$

Algorithm 1 Robust Q-Learning

Initialization: T, $Q_0(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, behavior policy π_b, s_0, step size α_t

for $t = 0, 1, 2, ..., T-1$ **do**
  Choose a_t according to $\pi_b(\cdot|s_t)$
  Observe s_{t+1} and c_t
  $V_t(s) \leftarrow \min_{a\in\mathcal{A}} Q_t(s,a), \forall s \in \mathcal{S}$
  $Q_{t+1}(s_t,a_t) \leftarrow (1-\alpha_t)Q_t(s_t,a_t) + \alpha_t(c_t + \gamma\sigma_{\hat{\mathcal{P}}_t}(V_t))$
  $Q_{t+1}(s,a) \leftarrow Q_t(s,a)$ for $(s,a) \neq (s_t,a_t)$
end for

Output: Q_T

Compared to the model-based approach, our approach is model-free. It does not require the prior knowledge of the uncertainty set, i.e., the knowledge of $p_s^a, \forall s \in \mathcal{S}, a \in \mathcal{A}$. Furthermore, the memory requirement of our algorithm is $|\mathcal{S}| \times |\mathcal{A}|$ (used to store the Q-table), and unlike the model-based approach it does not need a table of size $|\mathcal{S}|^2|\mathcal{A}|$ to store $p_s^a, \forall s \in \mathcal{S}, a \in \mathcal{A}$, which could be problematic if the state space is large. Moreover, our algorithm does not involve a relaxation of the uncertainty set like the one in [48], which will incur a stringent condition on the discount factor to guarantee the convergence. As will be shown below, the convergence of our Algorithm 1 does not require any condition on the discount factor.

We show in the following theorem that the robust Q-learning algorithm converges asymptotically to the optimal robust action-value function Q^*.

Theorem 2. *(Asymptotic Convergence) If step sizes α_t satisfy that $\sum_{t=0}^{\infty} \alpha_t = \infty$ and $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$, then $Q_t \to Q^*$ as $t \to \infty$, with probability 1.*

To further establish the finite-time error bound for our robust Q-learning algorithm in Algorithm 1, we make the following assumption that is commonly used in the analysis of vanilla Q-learning.

Assumption 1. *The Markov chain induced by the behavior policy π_b and the transition kernel $p_s^a, \forall s \in \mathcal{S}, a \in \mathcal{A}$ is uniformly ergodic.*

Let μ_{π_b} denote the stationary distribution over $\mathcal{S} \times \mathcal{A}$ induced by π_b and $p_s^a, \forall s \in \mathcal{S}, a \in \mathcal{A}$. We then further define $\mu_{\min} = \min_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mu_{\pi_b}(s,a)$. This quantity characterizes how many samples are needed to visit every state-action pair sufficiently often. Define the following mixing time of the induced Markov chain:

$$t_{\text{mix}} = \min\left\{t : \max_{s\in\mathcal{S}} d_{\text{TV}}(\mu_\pi, P(s_t = \cdot|s_0 = s)) \leq \frac{1}{4}\right\}, \tag{7}$$

where d_{TV} is the total variation distance.

The following theorem establishes the finite-time error bound of our robust Q-learning algorithm.

Theorem 3. *(Finite-Time Error Bound) There exist some positive constants c_0 and c_1 such that for any $\delta < 1$, any $\epsilon < \frac{1}{1-\gamma}$, any T satisfying*

$$T \geq c_0\left(\frac{1}{\mu_{min}(1-\gamma)^5\epsilon^2} + \frac{t_{mix}}{\mu_{min}(1-\gamma)}\right)\log\left(\frac{T|\mathcal{S}||\mathcal{A}|}{\delta}\right)\log\left(\frac{1}{\epsilon(1-\gamma)^2}\right), \tag{8}$$

and step size

$$\alpha_t = \frac{c_1}{\log\left(\frac{T|\mathcal{S}||\mathcal{A}|}{\delta}\right)}\min\left(\frac{1}{t_{mix}}, \frac{\epsilon^2(1-\gamma)^4}{\gamma^2}\right), \forall t \geq 0 \tag{9}$$

we have with probability at least $1-6\delta$,

$$\|Q_T - Q^*\|_\infty \leq 3\epsilon. \tag{10}$$

From the theorem, we can see that to guarantee an ϵ-accurate robust Q-function estimate, a sample size of the order $\tilde{\mathcal{O}}(\frac{1}{\mu_{\min}(1-\gamma)^5\epsilon^2} + \frac{t_{\text{mix}}}{\mu_{\min}(1-\gamma)})$ (up to some logarithmic terms) is needed. This complexity matches with the one for the vanilla Q-learning in [30] (within a constant factor), while our algorithm also guarantees robustness to MDP model uncertainty. Our algorithm design and analysis can be readily extended to robustify TD and SARSA. The variance-reduction technique [58] can also be combined with our robust Q-learning algorithm to further improve the dependency on $(1-\gamma)$.

5 Function Approximation: Robust TDC

In this section, we investigate the case where the state and action spaces can be large or even continuous. A popular approach is to approximate the value function using a parameterized function, e.g., linear function and neural network. In this section, we focus on the case with linear function approximation to illustrate the main idea of designing robust RL algorithms. Our approach can be extended to non-linear (smooth) function approximation using techniques in, e.g., [13, 10, 57, 61].

We focus on the problem of robust policy evaluation, i.e., estimate the robust value function V^π defined in (1) for a given policy π under the worst-case MDP transition kernel in the uncertainty set. Note that for robust RL with model uncertainty, any policy evaluation problem can be viewed as "off-transition-kernel", as it is to evaluate the value function under the worst-case MDP using samples from a different MDP. Since the TD algorithm with function approximation may diverge under off-policy training [6] and importance sampling cannot be applied here due to unknown transition kernel, in this paper we generalize the GTD method [37, 36] to the robust setting.

Let $\left\{\phi^{(i)}:\mathcal{S}\to\mathbb{R},\, i=1,\ldots,N\right\}$ be a set of N fixed base functions, where $N \ll |\mathcal{S}||\mathcal{A}|$. In particular, we approximate the robust value function using a linear combination of $\phi^{(i)}$'s: $V_\theta(s) = \sum_{i=1}^N \theta^i \phi_s^{(i)} = \phi_s^\top \theta$, where $\theta \in \mathbb{R}^N$ is the weight vector.

Define the following robust Bellman operator for a given policy π:

$$\begin{aligned}\mathbf{T}_\pi V(s) &\triangleq \mathbb{E}_{A\sim\pi(\cdot|s)}[c(s,A)+\gamma\sigma_{\mathcal{P}_s^A}(V)]\\ &= \mathbb{E}_{A\sim\pi(\cdot|s)}\left[c(s,A)+\gamma(1-R)\sum_{s'\in\mathcal{S}} p_{s,s'}^A V(s') + \gamma R \max_{s'\in\mathcal{S}} V(s')\right]. \end{aligned} \tag{11}$$

We then define the mean squared projected robust Bellman error (MSPRBE) as

$$\text{MSPRBE}(\theta) = \|\mathbf{\Pi}\mathbf{T}_\pi V_\theta - V_\theta\|_{\mu_\pi}^2, \tag{12}$$

where $\|v\|_{\mu_\pi}^2 = \int v^2(s)\mu_\pi(ds)$, μ_π is the stationary distribution induced by π, and $\mathbf{\Pi}$ is a projection onto the linear function space w.r.t. $\|\cdot\|_{\mu_\pi}$. We will develop a two time-scale gradient-based approach to minimize the MSPRBE. However, it can be seen that $\max_s V_\theta(s)$ in (11) is not smooth in θ, which is troublesome in both algorithm design and analysis. To solve this issue, we introduce the following smoothed robust Bellman operator $\hat{\mathbf{T}}_\pi$ by smoothing the max with a softmax:

$$\hat{\mathbf{T}}_\pi V(s) = \mathbb{E}_{A\sim\pi(\cdot|s)}\left[c(s,A)+\gamma(1-R)\sum_{s'\in\mathcal{S}} p_{s,s'}^A V(s') + \gamma R \sum_{s'\in\mathcal{S}} p_V(s')V(s')\right], \tag{13}$$

where $p_V(s) = \frac{e^{\varrho V(s)}}{\sum_{s'\in\mathcal{S}} e^{\varrho V(s')}}$ is the softmax distribution w.r.t. V with a parameter $\varrho > 0$. Note that when $\varrho \to \infty$, the smoothed robust Bellman operator $\hat{\mathbf{T}}_\pi \to \mathbf{T}_\pi$. The softmax operator can also be replaced by some other operator that approximates the max operator and is smooth, e.g., mellow-max [3]. In the following, we show that the fixed point of $\hat{\mathbf{T}}_\pi$ converges to the one of $\mathbf{T}_\pi$ for large ϱ.

Theorem 4. *Let V_1 and V_2 be the fixed points of $\hat{\mathbf{T}}_\pi$ and $\mathbf{T}_\pi$, respectively. Then*

$$\|V_1 - V_2\|_\infty \le \frac{2\|V_2\|_\infty \gamma R}{1-\gamma} d_{TV}(\mathbb{1}_{\arg\max V_2}, p_{V_2})) \to 0, \text{ as } \varrho\to\infty, \tag{14}$$

where $\mathbb{1}_{\arg\max V_2}$ is the distribution over $\mathcal{S}$ that has probability 1 at $\arg\max_{s\in\mathcal{S}} V_2(s)$.

We then denote by $J(\theta)$ the smoothed MSPRBE with the softmax operator, and the goal is to minimize it:

$$\min_\theta J(\theta) = \min_\theta \left\|\mathbf{\Pi}\hat{\mathbf{T}}_\pi V_\theta - V_\theta\right\|_{\mu_\pi}^2. \tag{15}$$

5.1 Algorithm Development

In the following, we develop the robust TDC algorithm to solve the problem in (15). We will first derive the gradient of the smoothed MSPRBE, $J(\theta)$, and then design a two time-scale update rule

using the weight doubling trick in [53] to solve the double sampling problem. Define $\delta_{s,a,s'}(\theta) \triangleq c(s,a)+\gamma(1-R)V_\theta(s')+\gamma R\sum_{j\in\mathcal{S}} p_\theta(j)V_\theta(j)-V_\theta(s)$, where $p_\theta = p_{V_\theta}$ is the soft-max distribution w.r.t. $V_\theta = \theta^\top\phi$. Denote by $C \triangleq \mathbb{E}_{\mu_\pi}\left[\phi_S^\top\phi_S\right]$. Then, $\mathbb{E}_\mu[\delta_{S,A,S'}(\theta)\phi_S] = \Phi^\top D\left(\hat{\mathbf{T}}_\pi V_\theta - V_\theta\right)$, where $D = \text{diag}(\mu_\pi(s_1), \mu_\pi(s_2), ..., \mu_\pi(s_{|\mathcal{S}|}))$ and $\Phi = (\phi_{s_1}, \phi_{s_2}, ..., \phi_{s_{|\mathcal{S}|}})^\top \in \mathbb{R}^{|\mathcal{S}|\times N}$. We know that $\mathbf{\Pi}^\top D\mathbf{\Pi} = D^\top\Phi(\Phi^\top D\Phi)^{-1}\Phi^\top D$. Then, we have

$$J(\theta) = \left\|\mathbf{\Pi}\hat{\mathbf{T}}_\pi V_\theta - V_\theta\right\|_{\mu_\pi}^2 = \mathbb{E}_{\mu_\pi}[\delta_{S,A,S'}(\theta)\phi_S]^\top C^{-1}\mathbb{E}_{\mu_\pi}[\delta_{S,A,S'}(\theta)\phi_S]. \tag{16}$$

Then, its gradient can be written as:

$$\begin{aligned}
-\frac{1}{2}\nabla J(\theta) &= -\mathbb{E}_{\mu_\pi}[(\nabla\delta_{S,A,S'}(\theta))\phi_S]^\top C^{-1}\mathbb{E}_{\mu_\pi}[\delta_{S,A,S'}(\theta)\phi_S] \\
&= \mathbb{E}_{\mu_\pi}[\delta_{S,A,S'}(\theta)\phi_S] - \gamma\mathbb{E}_{\mu_\pi}\left[\left((1-R)\phi_{S'} + R\sum_{s\in\mathcal{S}}(\nabla p_\theta(s)V_\theta(s) + p_\theta(s)\phi_s)\right)\phi_S^\top\right]\omega(\theta),
\end{aligned}$$

where $\omega(\theta) = C^{-1}\mathbb{E}_{\mu_\pi}[\delta_{S,A,S'}(\theta)\phi_S]$. It can be seen that to obtain an unbiased estimate of $\nabla J(\theta)$, two independent samples are needed as there exists a multiplication of two expectations, which is not applicable when there is only one sample trajectory. We then utilize the weight doubling trick in [53], and design the robust TDC algorithm in Algorithm 2. Specifically, we introduce a fast time scale to estimate $\omega(\theta)$, and a slow time scale to estimate $\nabla J(\theta)$. Denote the projection by $\mathbf{\Pi}_K(x) \triangleq \arg\min_{\|y\|\le K}\|y-x\|$ for any $x \in \mathbb{R}^N$. Our robust TDC algorithm in Algorithm 2 can be implemented in an online and incremental fashion. If the uncertainty set becomes a singleton, i.e., $R = 0$, then Algorithm 2 reduces to the vanilla TDC algorithm.

Algorithm 2 Robust TDC with Linear Function Approximation

Input: T,α, β, ϱ, ϕ_i for $i = 1, ..., N$, projection radius K
Initialization: θ_0,w_0, s_0

1: Choose $W \sim \text{Uniform}(0, 1, ..., T-1)$
2: **for** $t = 0, 1, 2, ..., W-1$ **do**
3: Take action according to $\pi(\cdot|s_t)$ and observe s_{t+1} and c_t
4: $\phi_t \leftarrow \phi_{s_t}$
5: $p_{\theta_t}(s) \leftarrow \frac{e^{\varrho V_{\theta_t}(s)}}{\sum_{s'\in\mathcal{S}} e^{\varrho V_{\theta_t}(s')}}$ for all $s \in \mathcal{S}$
6: $\delta_t(\theta_t) \leftarrow c_t + \gamma(1-R)V_{\theta_t}(s_{t+1}) + \gamma R\sum_{s\in\mathcal{S}} p_{\theta_t}(s)V_{\theta_t}(s) - V_{\theta_t}(s_t)$
7: $\theta_{t+1} \leftarrow \mathbf{\Pi}_K\left(\theta_t + \alpha\left(\delta_t(\theta_t)\phi_t - \gamma\left((1-R)\phi_{t+1} + R\sum_{s\in\mathcal{S}}(\nabla p_{\theta_t}(s)V_{\theta_t}(s) + p_{\theta_t}(s)\phi_s)\right)\phi_t^\top\omega_t\right)\right)$
8: $\omega_{t+1} \leftarrow \mathbf{\Pi}_K(\omega_t + \beta(\delta_t(\theta_t) - \phi_t^\top\omega_t)\phi_t)$
9: **end for**

Output: θ_W

5.2 Finite-Time Error Bound of Robust TDC

Unlike the vanilla TDC algorithm, $J(\theta)$ here is non-convex. Therefore, we are interested in the convergence to stationary points, i.e., the rate of $\|\nabla J(\theta)\| \to 0$. We first make some standard assumptions which are commonly used in RL algorithm analysis, e.g., [60, 26, 65, 50, 9].

Assumption 2 (Bounded feature)**.** $\|\phi_s\|_2 \le 1, \forall s \in \mathcal{S}$.

Assumption 3 (Bounded cost function)**.** $|c(s,a)| \le c_{\max}, \forall s \in \mathcal{S}$ *and* $a \in \mathcal{A}$.

Assumption 4 (Problem solvability)**.** *The matrix* $C = \mathbb{E}_{\mu_\pi}[\phi_S\phi_S^\top]$ *is non-singular with* $\lambda > 0$ *being its smallest eigenvalue.*

Assumption 5 (Geometric uniform ergodicity)**.** *There exist some constants* $m > 0$ *and* $\rho \in (0,1)$ *such that*

$$\max_{s\in\mathcal{S}} d_{TV}(\mathbb{P}(s_t|s_0 = s), \mu_\pi) \le m\rho^t,$$

for any $t > 0$.

In the following theorem, we characterize the finite-time error bound for the convergence of our robust TDC algorithm. Here we only provide the order of the bounds in terms of T. The explicit bounds can be found in (135) in Appendix D.3.

Theorem 5. *Consider the following step-sizes:* $\beta = \mathcal{O}\left(\frac{1}{T^b}\right)$, *and* $\alpha = \mathcal{O}\left(\frac{1}{T^a}\right)$, *where* $\frac{1}{2} < a \leq 1$ *and* $0 < b \leq a$. *Then we have that*

$$\mathbb{E}[\|\nabla J(\theta_W)\|^2] = \mathcal{O}\left(\frac{1}{T\alpha} + \alpha\log(1/\alpha) + \frac{1}{T\beta} + \beta\log(1/\beta)\right), \tag{17}$$

If we further let $a = b = 0.5$, *then*

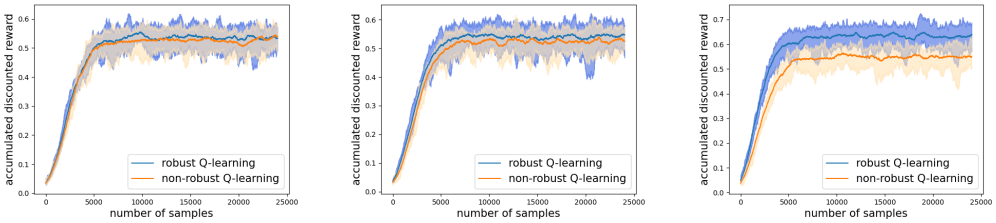
$$\mathbb{E}[\|\nabla J(\theta_W)\|^2] = \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right). \tag{18}$$

The robust TDC has a matching complexity with the vanilla TDC with non-linear function approximation [61], but provides the additional robustness to model uncertainty. It does not need to relax the uncertainty set like in [48], and our convergence results do not need a condition on the discount factor.

6 Experiments

6.1 Robust Q-Learning

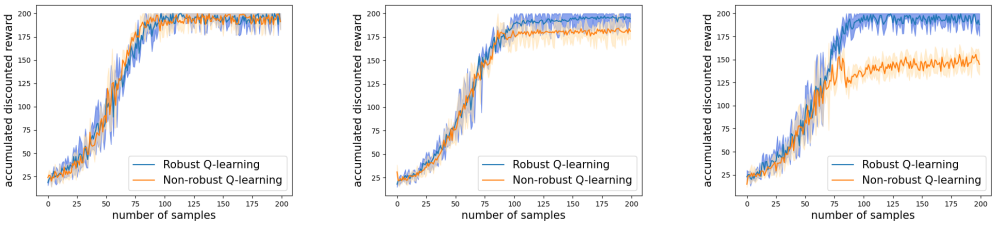
In this section, we compare our robust Q-learning with the vanilla non-robust Q-learning. We use OpenAI gym framework [12], and consider two different problems: Frozen lake and Cart-Pole. One more example of the taxi problem is given in the appendix. To demonstrate the robustness, the policy is learned in a perturbed MDP, and is then tested on the true unperturbed MDP. Specifically, during the training, we set a probability p such that after the agent takes an action, with probability p, the state transition is uniformly over $\mathcal{S}$, and with probability $1-p$ the state transition is according to the true unperturbed transition kernel. The behavior policy for all the experiments below is set to be a uniform distribution over the action space given any state, i.e., $\pi_b(a|s) = \frac{1}{|\mathcal{A}|}$ for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$. We then evaluate the performance of the obtained policy in the unperturbed environment. At each time t, the policy we evaluate is the greedy-policy w.r.t. the current estimate of the Q-function, i.e., $\pi_t(s) = \arg\max_a Q_t(s,a)$. A Monte-Carlo method with horizon 100 is used to evaluate the accumulated discounted reward of the learned policy on the unperturbed MDP. We take the average over 30 trajectories. More details about the experiment setup are provided in the appendix.



(a) p=0.1, R=0.1 (b) p=0.05, R=0.2 (c) p=0.1, R=0.2

Figure 1: **FrozenLake-v0**: robust Q-learning v.s. non-robust Q-learning.

In Figure 1 and Figure 2, we plot the accumulated discounted reward of both algorithms under different p and R for both problems. The upper and lower envelopes of the curves correspond to the 95 and 5 percentiles of the 30 trajectories, respectively. It can be seen that overall our robust Q-learning algorithm achieves a much higher reward than the vanilla Q-learning. This demonstrates the robustness of our robust Q-learning algorithm to model uncertainty. Moreover, as p and R getting larger, i.e., as the MDP that we learn the policy deviates from the MDP we test the policy, the advantage of our robust Q-learning algorithm is getting more significant compared to the vanilla Q-learning algorithm.



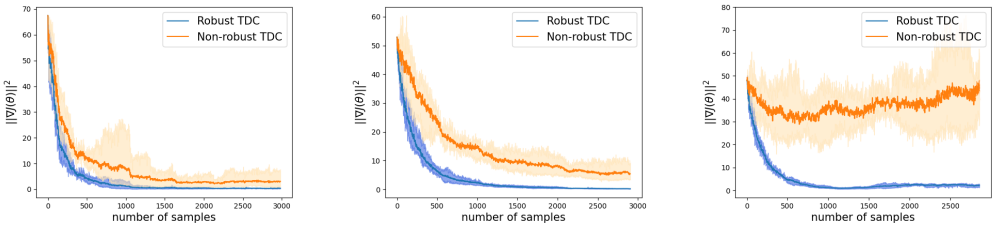
(a) p=0.1, R=0.1 (b) p=0.05, R=0.2 (c) p=0.1, R=0.2

Figure 2: **CartPole-v0**: robust Q-learning v.s. non-robust Q-learning.

6.2 Robust TDC with Linear Function Approximation

In this section we compare our robust TDC with the vanilla non-robust TDC with linear function approximation on the 4×4 Frozen Lake problem. The problem setting is the same as the one in Section 6.1. More details about the experiment setup are provided in the appendix.

We implement the two algorithms using samples from the perturbed MDP both for 30 times, and obtain 30 sequences of $\{\theta_t^i\}_{t=1}^{\infty}$, $i = 1, 2, ..., 30$. We then compute the squared gradient norm $\|\nabla J(\theta)\|^2$ on the true unperturbed MDP, and see whether $\{\theta_t^i\}_{t=1}^{\infty}$ converges to some stationary points on the true unperturbed MDP. In Fig. 3, we plot the average squared gradient norm $\|\nabla J(\theta)\|^2$ for different p and R. The upper and lower envelops are the 95 and 5 percentiles of the 30 curves. It can be seen that our robust TDC converges much faster than vanilla TDC, and as the model mismatch between the training and test MDPs enlarges, the vanilla TDC may diverge (Fig. 3(c)), while our robust TDC still converges to some stationary point. Also, the robust TDC has a much smaller variance, which indicates a much stable behavior under model uncertainty.



(a) p=0.1, R=0.1 (b) p=0.05, R=0.2 (c) p=0.1, R=0.2

Figure 3: **FrozenLake-v0**: $\|\nabla J(\theta)\|^2$ of robust TDC and non-robust TDC.

7 Conclusion

In this paper, we develop a novel approach for solving model-free robust RL problems with model uncertainty. Our algorithms can be implemented in an online and incremental fashion, do not require additional memory than their non-robust counterparts. We theoretically proved the convergence of our algorithms under no additional assumption on the discount factor, and further characterized their finite-time error bounds, which match with their non-robust counterparts (within a constant factor). Our approach can be readily extended to robustify TD, SARSA and other GTD algorithms. **Limitations:** It is also of future interest to investigate robustness to reward uncertainty, and other types of uncertainty sets, e.g., ones defined by KL divergence, Wasserstein distance and total variation. **Negative societal impact:** To the best of the authors' knowledge, this study does not have any potential negative impact on the society.

References

[1] M. A. Abdullah, H. Ren, H. B. Ammar, V. Milenkovic, R. Luo, M. Zhang, and J. Wang. Wasserstein robust reinforcement learning. *arXiv preprint arXiv:1907.13196*, 2019.

[2] A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *arXiv preprint arXiv:1908.00261*, 2019.

[3] 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.

[4] C. G. Atkeson and J. Morimoto. Nonparametric representation of policies and value functions: A trajectory-based approach. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1643–1650, 2003.

[5] J. A. Bagnell, A. Y. Ng, and J. G. Schneider. Solving uncertain Markov decision processes. 2001.

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

[7] C. L. Beck and R. Srikant. Error bounds for constant step-size Q-learning. *Systems & control letters*, 61(12):1203–1208, 2012.

[8] J. Bhandari and D. Russo. Global optimality guarantees for policy gradient methods. *arXiv preprint arXiv:1906.01786*, 2019.

[9] J. Bhandari, D. Russo, and R. Singal. A finite time analysis of temporal difference learning with linear function approximation. In *Proc. Annual Conference on Learning Theory (CoLT)*, pages 1691–1692. PMLR, 2018.

[10] S. Bhatnagar, D. Precup, D. Silver, R. S. Sutton, H. Maei, and C. Szepesvári. Convergent temporal-difference learning with arbitrary smooth function approximation. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, volume 22, pages 1204–1212, 2009.

[11] V. S. Borkar and S. P. Meyn. The ode method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, 38(2):447–469, 2000.

[12] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, 2016.

[13] 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.

[14] S. Cen, C. Cheng, Y. Chen, Y. Wei, and Y. Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *arXiv preprint arXiv:2007.06558*, 2020.

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

[16] G. Dalal, B. Szorenyi, and G. Thoppe. A tale of two-timescale reinforcement learning with the tightest finite-time bound. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, pages 3701–3708, 2020.

[17] 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.

[18] 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.

[19] T. T. Doan. Finite-time analysis and restarting scheme for linear two-time-scale stochastic approximation. *arXiv preprint arXiv:1912.10583*, 2019.

[20] E. Even-Dar, Y. Mansour, and P. Bartlett. Learning rates for Q-learning. *Journal of machine learning Research*, 5(1), 2003.

[21] 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.

[22] L. Hou, L. Pang, X. Hong, Y. Lan, Z. Ma, and D. Yin. Robust reinforcement learning with wasserstein constraint. *arXiv preprint arXiv:2006.00945*, 2020.

[23] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

[24] P. J. Huber. A robust version of the probability ratio test. *Ann. Math. Statist.*, 36:1753–1758, 1965.

[25] G. N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.

[26] M. Kaledin, E. Moulines, A. Naumov, V. Tadic, and H.-T. Wai. Finite time analysis of linear two-timescale stochastic approximation with Markovian noise. In *Proc. Annual Conference on Learning Theory (CoLT)*, pages 2144–2203. PMLR, 2020.

[27] J. Kos and D. Song. Delving into adversarial attacks on deep policies. *arXiv preprint arXiv:1705.06452*, 2017.

[28] H. Kumar, A. Koppel, and A. Ribeiro. On the sample complexity of actor-critic method for reinforcement learning with function approximation. *arXiv preprint arXiv:1910.08412*, 2019.

[29] G. Li, C. Cai, Y. Chen, Y. Gu, Y. Wei, and Y. Chi. Is Q-learning minimax optimal? a tight sample complexity analysis. *arXiv preprint arXiv:2102.06548*, 2021.

[30] G. Li, Y. Wei, Y. Chi, Y. Gu, and Y. Chen. Sample complexity of asynchronous Q-learning: Sharper analysis and variance reduction. *arXiv preprint arXiv:2006.03041*, 2020.

[31] S. H. Lim, H. Xu, and S. Mannor. Reinforcement learning in robust Markov decision processes. *Proc. Advances in Neural Information Processing Systems (NIPS)*, 26:701–709, 2013.

[32] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun. Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748*, 2017.

[33] 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.

[34] S. Ma, Y. Zhou, and S. Zou. Variance-reduced off-policy TDC learning: Non-asymptotic convergence analysis. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 14796–14806, 2020.

[35] S. Ma, Y. Zhou, and S. Zou. Greedy-GQ with variance reduction: Finite-time analysis and improved complexity. In *Proc. International Conference on Learning Representations (ICLR)*, 2021.

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

[37] 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)*, pages 719–726, 2010.

[38] A. Mandlekar, Y. Zhu, A. Garg, L. Fei-Fei, and S. Savarese. Adversarially robust policy learning: Active construction of physically-plausible perturbations. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3932–3939. IEEE, 2017.

[39] J. Mei, C. Xiao, C. Szepesvari, and D. Schuurmans. On the global convergence rates of softmax policy gradient methods. In *Proc. International Conference on Machine Learning (ICML)*, pages 6820–6829. PMLR, 2020.

[40] J. Morimoto and K. Doya. Robust reinforcement learning. *Neural computation*, 17(2):335–359, 2005.

[41] A. Nilim and L. El Ghaoui. Robustness in Markov decision problems with uncertain transition matrices. In *NIPS*, pages 839–846. Citeseer, 2003.

[42] A. Nilim and L. El Ghaoui. Robustness in Markov decision problems with uncertain transition matrices. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 839–846, 2004.

[43] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary. Robust deep reinforcement learning with adversarial attacks. *arXiv preprint arXiv:1712.03632*, 2017.

[44] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta. Robust adversarial reinforcement learning. In *International Conference on Machine Learning*, pages 2817–2826. PMLR, 2017.

[45] S. Qiu, Z. Yang, J. Ye, and Z. Wang. On the finite-time convergence of actor-critic algorithm. In *Proc. Optimization Foundations for Reinforcement Learning Workshop at Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[46] G. Qu and A. Wierman. Finite-time analysis of asynchronous stochastic approximation and q-learning. In *Conference on Learning Theory*, pages 3185–3205. PMLR, 2020.

[47] A. Rajeswaran, S. Ghotra, B. Ravindran, and S. Levine. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*, 2016.

[48] A. Roy, H. Xu, and S. Pokutta. Reinforcement learning under model mismatch. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 3046–3055, 2017.

[49] J. K. Satia and R. E. Lave Jr. Markovian decision processes with uncertain transition probabilities. *Operations Research*, 21(3):728–740, 1973.

[50] R. Srikant and L. Ying. Finite-time error bounds for linear stochastic approximation and TD learning. In *Proc. Annual Conference on Learning Theory (CoLT)*, pages 2803–2830, Phoenix, USA, Jun 2019.

[51] J. Sun, G. Wang, G. B. Giannakis, Q. Yang, and Z. Yang. Finite-sample analysis of decentralized temporal-difference learning with linear function approximation. *arXiv preprint arXiv:1911.00934*, 2019.

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

[53] 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.

[54] R. S. Sutton, C. Szepesvári, and H. R. Maei. A convergent o (n) algorithm for off-policy temporal-difference learning with linear function approximation.

[55] A. Tamar, S. Mannor, and H. Xu. Scaling up robust MDPs using function approximation. In *Proc. International Conference on Machine Learning (ICML)*, pages 181–189, 2014.

[56] E. Vinitsky, Y. Du, K. Parvate, K. Jang, P. Abbeel, and A. Bayen. Robust reinforcement learning using adversarial populations. *arXiv preprint arXiv:2008.01825*, 2020.

[57] H.-T. Wai, M. Hong, Z. Yang, Z. Wang, and K. Tang. Variance reduced policy evaluation with smooth function approximation. *Advances in Neural Information Processing Systems*, 32:5784–5795, 2019.

[58] M. J. Wainwright. Variance-reduced q-learning is minimax optimal. *arXiv preprint arXiv:1906.04697*, 2019.

[59] L. Wang, Q. Cai, Z. Yang, and Z. Wang. Neural policy gradient methods: Global optimality and rates of convergence. *arXiv preprint arXiv:1909.01150*, 2019.

[60] Y. Wang and S. Zou. Finite-sample analysis of Greedy-GQ with linear function approximation under Markovian noise. In *Proc. International Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 11–20. PMLR, 2020.

[61] Y. Wang, S. Zou, and Y. Zhou. Finite-sample analysis for two time-scale non-linear TDC with general smooth function approximation. *arXiv preprint arXiv:2104.02836*, 2021.

[62] W. Wiesemann, D. Kuhn, and B. Rustem. Robust Markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013.

[63] Y. Wu, W. Zhang, P. Xu, and Q. Gu. A finite time analysis of two time-scale actor critic methods. *arXiv preprint arXiv:2005.01350*, 2020.

[64] T. Xu and Y. Liang. Sample complexity bounds for two timescale value-based reinforcement learning algorithms. *arXiv preprint arXiv:2011.05053*, 2020.

[65] 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.

[66] Z. Yang, Y. Chen, M. Hong, and Z. Wang. Provably global convergence of actor-critic: A case for linear quadratic regulator with ergodic cost. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 8353–8365, 2019.

[67] K. Zhang, B. Hu, and T. Basar. On the stability and convergence of robust adversarial reinforcement learning: A case study on linear quadratic systems. *Advances in Neural Information Processing Systems*, 33, 2020.

[68] K. Zhang, T. Sun, Y. Tao, S. Genc, S. Mallya, and T. Basar. Robust multi-agent reinforcement learning with model uncertainty. *Advances in Neural Information Processing Systems*, 33, 2020.

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

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We used the OpenAI gyme framework, and cited the paper.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]