Robust Reinforcement Learning using Least Squares Policy Iteration with Provable Performance Guarantees

Kishan Panaganti [1] **Dileep Kalathil** [1]

Abstract

This paper addresses the problem of model-free reinforcement learning for Robust Markov Decision Process (RMDP) with large state spaces. The goal of the RMDP framework is to find a policy that is robust against the parameter uncertainties due to the mismatch between the simulator model and real-world settings. We first propose the Robust Least Squares Policy Evaluation algorithm, which is a multi-step online model-free learning algorithm for policy evaluation. We prove the convergence of this algorithm using stochastic approximation techniques. We then propose Robust Least Squares Policy Iteration (RLSPI) algorithm for learning the optimal robust policy. We also give a general weighted Euclidean norm bound on the error (closeness to optimality) of the resulting policy. Finally, we demonstrate the performance of our RLSPI algorithm on some standard benchmark problems.

1. Introduction

Model-free Reinforcement Learning (RL) algorithms typically learn a policy by training on a simulator. In the RL literature, it is nominally assumed that the testing environment is identical to the training environment (simulator model). However, in reality, the parameters of the simulator model can be different from the real-world setting. This can be due to the approximation errors incurred while modeling, due to the changes in the real-world parameters over time, and can even be due to possible adversarial disturbances in the real-world. For example, in many robotics applications, the standard simulator parameter settings (mass, friction, wind conditions, sensor noise, action delays) can be different from that of the actual robot in the real-world. This mismatch between the training and testing environment parameters can significantly degrade the real-world performance of the model-free learning algorithms trained on a simulator model.

The RMDP framework (Iyengar, 2005; Nilim & El Ghaoui, 2005) addresses the *planning* problem of computing the optimal policy that is robust against parameter uncertainties that cause the mismatch between the training and testing environment parameters. The RMDP problem has been analyzed extensively in the tabular case (Iyengar, 2005; Nilim & El Ghaoui, 2005; Wiesemann et al., 2013; Xu & Mannor, 2010; Yu & Xu, 2015) and under the linear function approximation (Tamar et al., 2014). Algorithms for learning the optimal robust policy with provable guarantees have been proposed, both in the model-free (Roy et al., 2017) and model-based (Lim et al., 2013) reinforcement learning settings. However, the theoretical guarantees from these works are limited to the tabular RMDP settings. Learning policies for problems with large state spaces is computationally challenging. RL algorithms typically overcome this issue by using function approximation architectures, such as linear basis functions (Lagoudakis & Parr, 2003), reproducing kernel Hilbert spaces (RKHS) (Yang & Wang, 2020) and deep neural networks (Lillicrap et al., 2016). Recently, robust reinforcement learning problem has been addressed using deep RL methods (Pinto et al., 2017; Mankowitz et al., 2020; Zhang et al., 2020; Derman et al., 2018; Vinitsky et al., 2020). However, these works are empirical in nature and do not provide any theoretical guarantees for the learned policies. The problem of learning optimal robust policies with provable performance guarantees for RMDPs with large state spaces has not been well studied in the literature.

In this paper, we address the problem of learning a policy that is provably robust against the parameter uncertainties for RMDPs with large state spaces. In particular, we propose an online model-free reinforcement learning algorithm with linear function approximation for learning the optimal robust policy, and provide theoretical guarantees on the performance of the learned policy. Our choice of linear function approximation is motivated by its analytical tractability while providing the scaling to large state spaces. Indeed, linear function approximation based approaches have been successful in providing algorithms with provable

[1]Department of Electrical and Computer Engineering, Texas A&M University, College Station, United States. Correspondence to: Kishan Panaganti <kpb@tamu.edu>, Dileep Kalathil <dileep.kalathil@tamu.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

guarantees for many challenging problems in RL, including online model-free exploration (Jin et al., 2020; Yang & Wang, 2020), imitation learning (Abbeel & Ng, 2004; Arora et al., 2020), meta reinforcement learning (Wang et al., 2020; Kong et al., 2020), and offline reinforcement learning (Wang et al., 2021; Duan et al., 2020). Robust RL is much more challenging than the standard (non-robust) RL problems due to the inherent nonlinearity associated with the robust dynamic programming. We overcome this issue by a cleverly designed approximate dynamic programming approach. We then propose a model-free robust policy iteration using this approach with provable guarantees. Our algorithmic and technical contributions are as follows:

(i) Robust Least Squares Policy Evaluation (RLSPE(λ)) algorithm: A learning-based policy iteration algorithm needs to learn the value of a policy for performing (greedy) policy improvement. For this, we first propose RLSPE(λ) algorithm, a multi-step, online, model-free policy evaluation algorithm with linear function approximation. This can be thought as the robust version of classical least squares based RL algorithms for policy evaluation, like LSTD(λ) and LSPE(λ). We prove the convergence of this algorithm using stochastic approximation techniques, and also characterize its approximation error due to the linear architecture.

(ii) Robust Least Squares Policy Iteration (RLSPI) algorithm: We propose the RLSPI algorithm for learning the optimal robust policy. We also give a general L_2-norm bound on the error (closeness to optimality) of the resulting policy at any iterate of the algorithm. To the best of our knowledge, this is the first work that presents a learning based policy iteration algorithm for robust reinforcement learning with such provable guarantees.

(iii) Finally, we demonstrate the performance of the RLSPI algorithm on various standard RL test environments.

1.1. Related Work

RMDP formulation to address the parameter uncertainty problem was first proposed by (Iyengar, 2005) and (Nilim & El Ghaoui, 2005). (Iyengar, 2005) showed that the optimal robust value function and policy can be computed using the robust counterparts of the standard value iteration and policy iteration. To tackle the parameter uncertainty problem, other works considered distributionally robust setting (Xu & Mannor, 2010), modified policy iteration (Kaufman & Schaefer, 2013), and more general uncertainty set (Wiesemann et al., 2013). We note that the focus of these works were mainly on the planning problem in the tabular setting. Linear function approximation method to solve large RMDPs was proposed in (Tamar et al., 2014). Though this work suggests a sampling based approach, a general model-free learning algorithm and analysis was not included. (Roy et al., 2017) proposed the robust versions of the classical model-free reinforcement learning algorithms such as Q-learning, SARSA, and TD-learning in the tabular setting. They also proposed function approximation based algorithms for the policy evaluation. However, this work does not have a policy iteration algorithm with provable guarantees for learning the optimal robust policy. (Derman et al., 2018) introduced soft-robust actor-critic algorithms using neural networks, but does not provide any global convergence guarantees for the learned policy. (Tessler et al., 2019) proposed a min-max game framework to address the robust learning problem focusing on the tabular setting. (Lim & Autef, 2019) proposed a kernel-based RL algorithm for finding the robust value function in a batch learning setting. (Mankowitz et al., 2020) employed an entropy-regularized policy optimization algorithm for continuous control using neural network, but does not provide any provable guarantees for the learned policy.

Our work differs from the above in two significant ways. Firstly, we develop a new multi-step model-free reinforcement learning algorithm, RLSPE(λ), for policy evaluation. Extending the classical least squares based policy evaluation algorithms, like LSPE(λ) and LSTD(λ) (Bertsekas & Ioffe, 1996; Nedić & Bertsekas, 2003; Bertsekas, 2012), to the robust case is very challenging due to the nonlinearity of the robust TD(λ) operator. We overcome this issue by a cleverly defined approximate robust TD(λ) operator that is amenable to online learning using least squares approaches. Also, as pointed out in (Bertsekas, 2011), convergence analysis of least squares style algorithms for RL is different from that of the standard temporal difference (TD) algorithm. Secondly, we develop a new robust policy iteration algorithm with provable guarantees on the performance of the policy at any iterate. In particular, we give a general weighted Euclidean norm bound on the error of the resulting policy. While similar results are available for the non-robust settings, this is the first work to provide such a characterization in the challenging setting of robust reinforcement learning.

2. Background and Problem Formulation

A Markov Decision Process is a tuple $M = (\mathcal{S}, \mathcal{A}, r, P, \alpha)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\alpha \in (0, 1)$ is the discount factor. The transition probability matrix $P_{s,a}(s')$ represents the probability of transitioning to state s' when action a is taken at state s. We consider a finite MDP setting where the cardinality of state and action spaces are finite (but very large). A (deterministic) policy π maps each state to an action. The value of a policy π evaluated at state s is given by

$$V_{\pi,P}(s) = \mathbb{E}_{\pi,P}[\sum_{t=0}^{\infty} \alpha^t r(s_t, a_t) \mid s_0 = s],$$

where $a_t \sim \pi(s_t)$ and $s_{t+1} \sim P_{s_t,a_t}(\cdot)$. The optimal value function and the optimal policy of an MDP with the transition probability P are defined as $V_P^* = \max_\pi V_{\pi,P}$ and $\pi_P^* = \arg\max_\pi V_{\pi,P}$.

The RMDP formulation considers a set of model parameters (uncertainty set) under the assumption that the actual parameters lie in this uncertainty set, and the algorithm computes a robust policy that performs best under the worst model. More precisely, instead of a fixed transition probability matrix P, we consider a set of transition probability matrices $\mathcal{P}$. We assume that the set $\mathcal{P}$ satisfies the standard *rectangularity condition* (Iyengar, 2005). The objective is to find a policy that maximizes the worst-case performance. Formally, the *robust value function* V_π corresponding to a policy π and the *optimal robust value function* V^* are defined as (Iyengar, 2005; Nilim & El Ghaoui, 2005)

$$V_\pi = \inf_{P\in\mathcal{P}} V_{\pi,P}, \quad V^* = \sup_\pi \inf_{P\in\mathcal{P}} V_{\pi,P}. \tag{1}$$

The *optimal robust policy* π^* is such that the robust value function corresponding to it matches the optimal robust value function, that is, $V_{\pi^*} = V^*$.

A generic characterization of the set $\mathcal{P}$ makes the RMDPs problems intractable to solve by model-free methods. In the standard model-free methods, the algorithm has access to a simulator that can simulate the next state given the current state and current action, according to a fixed transition probability matrix (that is unknown to the algorithm). However, generating samples according to each and every transition probability matrix from the set $\mathcal{P}$ is clearly infeasible. To overcome this difficulty, we use the characterization of the uncertainty set used in (Roy et al., 2017).

Assumption 1 (Uncertainty Set)**.** *Each $P \in \mathcal{P}$ can be represented as $P_{s,a}(\cdot) = P^o_{s,a}(\cdot) + U_{s,a}(\cdot)$ for some $U_{s,a} \in \mathcal{U}_{s,a}$, where $P^o_{s,a}(\cdot)$ is the unknown transition probability matrix corresponding to the nominal (simulator) model and $\mathcal{U}_{s,a}$ is a confidence region around it.*

Using the above characterization, we can write $\mathcal{P} = \{P^o + U : U \in \mathcal{U}\}$, where $\mathcal{U} = \cup_{s,a}\mathcal{U}_{s,a}$. So, $\mathcal{U}$ is the set of all possible perturbations to the nominal model P^o.

An example of the uncertainty set $\mathcal{U}$ can be the spherical uncertainty set with a radius parameter. Define $\mathcal{U}_{s,a} := \{x \mid \|x\|_2 \le r, \sum_{s\in\mathcal{S}} x_s = 0, -P^o_{s,a}(s') \le x_{s'} \le 1 - P^o_{s,a}(s'), \forall s' \in \mathcal{S}\}$, for all $(s,a) \in (\mathcal{S},\mathcal{A})$, for some $r > 0$. Notice that, this uncertainty set uses the knowledge of the nominal model P^o in its construction. In practice, we do not know P^o. So, in Section 3.2, we introduce an approximate uncertainty set without using this information.

We consider *robust Bellman operator for policy evaluation*, defined as (Iyengar, 2005)

$$T_\pi(V)(s) = r(s,\pi(s)) + \alpha \inf_{P\in\mathcal{P}} \sum_{s'} P_{s,\pi(s)}(s')V(s'), \tag{2}$$

a popular approach to solve (1). Using our characterization of the uncertainty set, we can rewrite (2) as

$$T_\pi(V)(s) = r(s,\pi(s)) + \alpha \sum_{s'} P^o_{s,\pi(s)}(s')V(s') + \alpha \inf_{U\in\mathcal{U}_{s,\pi(s)}} \sum_{s'} U_{s,\pi(s)}(s')V(s'). \tag{3}$$

For any set $\mathcal{B}$ and a vector v, define $\sigma_\mathcal{B}(v) = \inf\{u^\top v : u \in \mathcal{B}\}$. We denote $|\mathcal{S}|$ as the cardinality of the set $\mathcal{S}$. Let $\sigma_{\mathcal{U}_\pi}(v)$ and r_π be the $|\mathcal{S}|$ dimensional column vectors defined as $(\sigma_{\mathcal{U}_{s,\pi(s)}}(v) : s \in \mathcal{S})^\top$ and $(r(s,\pi(s)) : s \in \mathcal{S})^\top$, respectively. Let P^o_π be the stochastic matrix corresponding to the policy π where for any $s, s' \in \mathcal{S}$, $P^o_\pi(s,s') = P^o_{s,\pi(s)}(s')$. Then, (3) can be written in the matrix form as

$$T_\pi(V) = r_\pi + \alpha P^o_\pi V + \alpha\sigma_{U_\pi}(V). \tag{4}$$

It is known (Iyengar, 2005) that T_π is a contraction in sup norm and the robust value function V_π is the unique fixed point of T_π. The *robust Bellman operator* T can also be defined in the same way as in the non-robust setting,

$$T(V) = \max_\pi T_\pi(V). \tag{5}$$

It is also known (Iyengar, 2005) that T is a contraction in sup norm, and the optimal robust value function V^* is its unique fixed point.

The goal of the robust RL is to learn the optimal robust policy π^* without knowing the nominal model P^o or the uncertainty set $\mathcal{P}$.

3. Robust Least Squares Policy Evaluation

In this section, we develop the RLSPE(λ) algorithm for learning the robust value function.

3.1. Robust TD(λ) Operator and the Challenges

In RL, a very useful approach for analyzing the multi-step learning algorithms like TD(λ), LSTD(λ), and LSPE(λ) is to define a multi-step Bellman operator called TD(λ) operator (Tsitsiklis & Van Roy, 1997; Bertsekas, 2012). Following the same approach, we can define the robust TD(λ) operator as well. For a given policy π, and a parameter $\lambda \in [0,1)$, the robust TD(λ) operator denoted by $T^{(\lambda)}_\pi : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ is defined as

$$T^{(\lambda)}_\pi(V) = (1-\lambda)\sum_{m=0}^{\infty} \lambda^m T^{m+1}_\pi(V). \tag{6}$$

Note that for $\lambda = 0$, we recover T_π. The following result is straightforward.

Proposition 1 (informal). *$T_\pi^{(\lambda)}$ is a contraction in sup norm and the robust value function V_π is its unique fixed point, for any $\alpha \in (0,1), \lambda \in [0,1)$.*

For RMDPs with very large state space, exact dynamic programming methods which involve the evaluation of (3) or (6) are intractable. A standard approach to overcome this issue is to approximate the value function using some function approximation architecture. Here we focus on linear function approximation architectures (Bertsekas, 2012). In linear function approximation architectures, the value function is represented as the weighted sum of features as, $\bar{V}(s) = \phi(s)^\top w, \forall s \in \mathcal{S}$, where $\phi(s) = (\phi_1(s), \phi_2(s), \ldots, \phi_L(s))^\top$ is an L dimensional feature vector with $L < |\mathcal{S}|$, and $w = (w_1, \cdots, w_L)^\top$ is a weight vector. In the matrix form, this can be written as $\bar{V} = \Phi w$ where Φ is an $|\mathcal{S}| \times L$ dimensional feature matrix whose s^{th} row is $\phi(s)^\top$. We assume linearly independent columns for Φ, i.e., rank(Φ) = L.

The standard approach to find an approximate (robust) value function is to solve for a w_π, with $\bar{V}_\pi = \Phi w_\pi$, such that $\Phi w_\pi = \Pi T_\pi^{(\lambda)} \Phi w_\pi$, where Π is a projection onto the subspace spanned by the columns of Φ. The projection is with respect to a d-weighted Euclidean norm. This norm is defined as $\|V\|_d^2 = V^\top D V$, where D is a diagonal matrix with non-negative diagonal entries $(d(s), s \in \mathcal{S})$, for any vector V. Under suitable assumptions, (Tamar et al., 2014) showed that ΠT_π is a contraction in a d-weighted Euclidean norm. We also use a similar assumption stated below.

Assumption 2. *(i) For any given policy π, there exists an exploration policy $\pi_e = \pi_{exp}(\pi)$ and a $\beta \in (0,1)$ such that $\alpha P_{s,\pi(s)}(s') \leq \beta P^o_{s,\pi_e(s)}(s')$, for all transition probability matrices $P \in \mathcal{P}$ and for all states $s, s' \in \mathcal{S}$.*
(ii) There exists a steady state distribution $d_{\pi_e} = (d_{\pi_e}(s), s \in \mathcal{S})$ for the Markov chain with transition probability $P^o_{\pi_e}$ with $d_{\pi_e}(s) > 0, \forall s \in \mathcal{S}$.

In the following, we will simply use d instead of d_{π_e}.

Though the above assumption appears restrictive, it is necessary to show that ΠT_π is a contraction in the d-weighted Euclidean norm, as proved in (Tamar et al., 2014). Also, a similar assumption is used in proving the convergence of off-policy reinforcement learning algorithm (Bertsekas & Yu, 2009). In the robust case, we can expect a similar condition because we are learning a robust value function for a set of transition probability matrices instead of a single transition probability matrix. We can now show the following.

Proposition 2 (informal). *Under Assumption 2, $\Pi T_\pi^{(\lambda)}$ is a contraction mapping in the d-weighted Euclidean norm for any $\lambda \in [0,1)$.*

The linear approximation based robust value function $\bar{V}_\pi = \Phi w_\pi$ can be computed using the iteration, $\Phi w_{k+1} = \Pi T_\pi^{(\lambda)} \Phi w_k$. Since $\Pi T_\pi^{(\lambda)}$ is a contraction, w_k will converge to w^*. A closed form solution for w_{k+1} given w_k can be found by **least squares approach** as $w_{k+1} = \arg\min_w \|\Phi w - \Pi T_\pi^{(\lambda)} \Phi w_k\|_d^2$. It can be shown that (details are given in the supplementary material), we can get a closed form solution for w_{k+1} as

$$w_{k+1} = w_k + (\Phi^\top D \Phi)^{-1} \Phi^\top D (T_\pi^{(\lambda)} \Phi w_k - \Phi w_k). \quad (7)$$

This is similar to the projected equation approach (Bertsekas, 2012) in the non-robust setting. Even in the non-robust setting, iterations using the (7) is intractable for MDPs with large state space. Moreover, when the transition matrix is unknown, it is not feasible to use (7) exactly even for small RMDPs. Simulation-based model-free learning algorithms are developed for addressing this problem in the non-robust case. In particular, LSPE(λ) algorithm (Nedić & Bertsekas, 2003; Bertsekas, 2012) is used to solve the iterations of the above form.

However, compared to the non-robust setting, there are two significant challenges in learning the robust value function by using simulation-based model-free approaches.

(i) Non-linearity of the robust TD(λ) operator: The non-robust T_π operator and the TD(λ) operator do not involve any nonlinear operations. So, they can be estimated efficiently from simulation samples in a model-free way. However, the robust TD(λ) operator when expanded will have the following form (derivation is given in the supplementary material).

$$T_\pi^{(\lambda)}(V) = (1-\lambda) \sum_{m=0}^{\infty} \lambda^m \Big(\sum_{k=0}^{m} (\alpha P_\pi^o)^k r_\pi + (\alpha P_\pi^o)^{m+1} V + \alpha \sum_{k=0}^{m} (\alpha P_\pi^o)^k \sigma_{\mathcal{U}_\pi}(T_\pi^{(m-k)} V) \Big). \quad (8)$$

The last term is very difficult to estimate using simulation-based model-free approaches due to the composition of operations $\sigma_{\mathcal{U}_\pi}$ and T_π. In addition, nonlinearity of the T_π operator by itself adds to the complexity.

(ii) Unknown uncertainty region $\mathcal{U}$: In our formulation, we assumed that the transition probability uncertainty set $\mathcal{P}$ is given by $\mathcal{P} = P^o + \mathcal{U}$. So, for each $U \in \mathcal{U}$, $P^o + U$ should be a valid transition probability matrix. However, in the model-free setting, we do not know the nominal transition probability P^o. So, it is not possible to know $\mathcal{U}$ exactly a priori. One can only use an approximation $\widehat{\mathcal{U}}$ instead of $\mathcal{U}$. This can possibly affect the convergence of the learning algorithms.

3.2. Robust Least Squares Policy Evaluation (RLSPE(λ)) Algorithm

We overcome the challenges of learning the robust value function by defining an approximate robust TD(λ) operator,

and by developing a robust least squares policy evaluation algorithm based on that.

Let $\widehat{\mathcal{U}}$ be the approximate uncertainty set we use instead of the actual uncertainty set. An example of the approximate uncertainty set $\widehat{\mathcal{U}}$ can be the spherical uncertainty set defined *without* using the knowledge of the model P^o as $\widehat{\mathcal{U}}_{s,a} := \{x \mid \|x\|_2 \leq r, \sum_{s\in\mathcal{S}} x_s = 0\}$ for all $(s,a) \in (\mathcal{S},\mathcal{A})$. Note that $P^o + U$ for $U \in \widehat{\mathcal{U}}$ need not be a valid transition probability matrix and this poses challenges both for the algorithm and analysis.

For a given policy π and a parameter $\lambda \in [0,1)$, approximate robust TD(λ) operator denoted by $\widetilde{T}_\pi^{(\lambda)} : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$, is defined as

$$\widetilde{T}_\pi^{(\lambda)}(V) = (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\left[\sum_{t=0}^{m}(\alpha P_\pi^o)^t r_\pi + \alpha\sum_{t=0}^{m}(\alpha P_\pi^o)^t\,\sigma_{\widehat{\mathcal{U}}_\pi}(V) + (\alpha P_\pi^o)^{m+1}V\right]. \quad (9)$$

Note that even with $\widehat{\mathcal{U}}_\pi = \mathcal{U}_\pi$, (9) is different from (8). We will show that this clever approximation helps to overcome the challenges due to the nonliterary associated with (8).

However, we emphasize that (9) is not an arbitrary definition. Note that, for $\widehat{\mathcal{U}}_\pi = \mathcal{U}_\pi$, with $\lambda = 0$, we recover the operator T_π. Moreover, the robust value function V_π is a fixed point of $\widetilde{T}_\pi^{(\lambda)}$ when $\widehat{\mathcal{U}}_\pi = \mathcal{U}_\pi$ for any $\lambda \in [0,1)$. We state this formally below.

Proposition 3. *Suppose $\widehat{\mathcal{U}}_\pi = \mathcal{U}_\pi$. Then, for any $\alpha \in (0,1)$ and $\lambda \in [0,1)$, the robust value function V_π is a fixed point of $\widetilde{T}_\pi^{(\lambda)}$, i.e., $\widetilde{T}_\pi^{(\lambda)}(V_\pi) = V_\pi$.*

Intuitively, the convergence of any learning algorithm using the approximate robust TD(λ) operator will depend on the difference between the actual uncertainty set $\mathcal{U}$ and its approximation $\widehat{\mathcal{U}}$. To quantify this, we use the following metric. Let $\rho = \max_{s\in\mathcal{S},a\in\mathcal{A}} \rho_{s,a}$ where

$$\rho_{s,a} = \max\begin{Bmatrix}\max_{x\in\widehat{\mathcal{U}}_{s,a}}\max_{y\in\mathcal{U}_{s,a}\setminus\widehat{\mathcal{U}}_{s,a}}\|x-y\|_d/d_{\min},\\ \max_{x\in\mathcal{U}_{s,a}}\max_{y\in\widehat{\mathcal{U}}_{s,a}\setminus\mathcal{U}_{s,a}}\|x-y\|_d/d_{\min}\end{Bmatrix}$$

and $d_{\min} := \min_{s\in\mathcal{S}} d(s)$. By convention, we set $\rho_{s,a} = 0$ when $\widehat{\mathcal{U}}_{s,a} = \mathcal{U}_{s,a}$ for all $(s,a) \in (\mathcal{S},\mathcal{A})$. So, $\rho = 0$ if $\widehat{\mathcal{U}} = \mathcal{U}$. Using this characterization and under some additional assumptions on the discount factor, we show that the approximate robust TD(λ) operator is a contraction in the d-weighted Euclidean norm.

Theorem 1. *Under Assumption 2, for any $V_1, V_2 \in \mathbb{R}^{|\mathcal{S}|}$ and $\lambda \in [0,1)$,*

$$\|\Pi\widetilde{T}_\pi^{(\lambda)}V_1 - \Pi\widetilde{T}_\pi^{(\lambda)}V_2\|_d \leq c(\alpha,\beta,\rho,\lambda)\,\|V_1 - V_2\|_d, \quad (10)$$

where $c(\alpha,\beta,\rho,\lambda) = (\beta(2-\lambda)+\rho\alpha)/(1-\beta\lambda)$. So, if $c(\alpha,\beta,\rho,\lambda) < 1$, $\Pi\widetilde{T}_\pi^{(\lambda)}$ is a contraction in the d-weighted Euclidean norm. Moreover, there exists a unique w_π such that $\Phi w_\pi = \Pi\widetilde{T}_\pi^{(\lambda)}(\Phi w_\pi)$. Furthermore, for this w_π,

$$\|V_\pi - \Phi w_\pi\|_d \leq \frac{1}{1-c(\alpha,\beta,\rho,\lambda)}\left(\|V_\pi - \Pi V_\pi\|_d + \frac{\beta\rho\|V_\pi\|_d}{1-\beta\lambda}\right). \quad (11)$$

We note that despite the assumption on the discount factor, we empirically show in Section 5 that our learning algorithm converges to a robust policy even if this assumption is violated. We also note that the upper bound in (11) quantifies the *error* of approximating the robust value function V_π with the approximate robust value function Φw_π. We will later use this error bound in in characterizing performance of both RLSPE and RLSPI algorithms.

Using the contraction property of approximate robust TD(λ) operator, the linear approximation based robust value function $\bar{V}_\pi = \Phi w_\pi$ can be computed using the iteration, $\Phi w_{k+1} = \Pi\widetilde{T}_\pi^{(\lambda)}\Phi w_k$. Similar to (7), we can get a closed form solution for w_{k+1} using **least squares approach** as

$$w_{k+1} = w_k + (\Phi^\top D\Phi)^{-1}\Phi^\top D(\widetilde{T}_\pi^{(\lambda)}\Phi w_k - \Phi w_k). \quad (12)$$

This can be written in a more succinct matrix form as given below (derivation is given in the supplementary material).

$$w_{k+1} = w_k + B^{-1}(Aw_k + C(w_k) + b), \text{ where,} \quad (13)$$

$$A = \Phi^\top D(\alpha P_\pi^o - I)\sum_{m=0}^{\infty}(\alpha\lambda P_\pi^o)^m\Phi, \quad (14)$$

$$B = \Phi^\top D\Phi, \quad (15)$$

$$C(w) = \alpha\Phi^\top D\sum_{t=0}^{\infty}(\alpha\lambda P_\pi^o)^t\sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w), \quad (16)$$

$$b = \Phi^\top D\sum_{t=0}^{\infty}(\alpha\lambda P_\pi^o)^t r_\pi. \quad (17)$$

Iterations by evaluating (13) exactly is intractable for MDPs with large state space, and infeasible if we do not know the transition probability P_π^o. To address this issue, we propose a simulation-based model-free online reinforcement learning algorithm, which we call robust least squares policy evaluation (RLSPE(λ)) algorithm, for learning the robust value function.

RLSPE(λ) algorithm: Generate a sequence of states and rewards, $(s_t, r_t, t \geq 0)$, using the policy π. Update the

parameters as

$$w_{t+1} = w_t + \gamma_t B_t^{-1}(A_t w_t + b_t + C_t(w_t)), \text{ where,} \quad (18)$$

$$A_t = \frac{1}{t+1}\sum_{\tau=0}^{t} z_\tau \left(\alpha\phi^\top(s_{\tau+1}) - \phi^\top(s_\tau)\right), \quad (19)$$

$$B_t = \frac{1}{t+1}\sum_{\tau=0}^{t} \phi(s_\tau)\phi^\top(s_\tau), \quad (20)$$

$$C_t(w) = \frac{\alpha}{t+1}\sum_{\tau=0}^{t} z_\tau \, \sigma_{\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}}(\Phi w), \quad (21)$$

$$b_t = \frac{1}{t+1}\sum_{\tau=0}^{t} z_\tau r(s_\tau, \pi(s_\tau)), \quad (22)$$

$$z_\tau = \sum_{m=0}^{\tau} (\alpha\lambda)^{\tau-m}\phi(s_m), \quad (23)$$

where γ_t is a deterministic sequence of step sizes. We assume that the step size satisfies the the standard Robbins-Munro stochastic conditions for stochastic approximation, i.e., $\sum_{t=0}^{\infty}\gamma_t = \infty$, $\sum_{t=0}^{\infty}\gamma_t^2 < \infty$.

We use the on-policy version of the RLSPE(λ) algorithm in the above description. So, we implicitly assume that the given policy π is an exploration policy according to the Assumption 2. This is mainly for the clarity of the presentation and notational convenience. Also, this simplifies the presentation of the policy iteration algorithm introduced in the next section. An off-policy version of the above algorithm can be implemented using the techniques given in (Bertsekas & Yu, 2009). We now give the convergence result of the RLSPE(λ) algorithm.

Theorem 2. *Let Assumption 2 hold. Also, let $c(\alpha,\beta,\rho,\lambda) < 1$ so that $\Pi\widetilde{T}_\pi^{(\lambda)}$ is a contraction according to Theorem 1. Let $\{w_t\}$ be the sequence generated by the RLSPE(λ) algorithm given in (18). Then, w_t converges to w_π with probability 1 where w_π satisfies the fixed point equation $\Phi w_\pi = \Pi\widetilde{T}_\pi^{(\lambda)}\Phi w_\pi$.*

The key idea of the proof is to show that the RLSPE(λ) update (18) approximates the exact update equation (12) and both converge to the same value w_π. One particularly challenging task is in analyzing the behavior of the term $C_t(w)$ due to the non-linearity of the function $\sigma_{\widehat{\mathcal{U}}_\pi}(.)$. We use the tools from stochastic approximation theory (Borkar, 2009; Nedić & Bertsekas, 2003) to show this rigorously after establishing the tractable properties of the function $\sigma_{\widehat{\mathcal{U}}_\pi}(.)$.

Note that Theorem 2 and Theorem 1 together give an error bound for the converged solution of the RLSPE(λ) algorithm. More precisely, Theorem 2 shows the convergence of the RLSPE(λ) algorithm to w_π and Theorem 1 gives the bound on $\|V_\pi - \Phi w_\pi\|_d$, which is the error due to linear function approximation. We will use this bound in the the convergence analysis of the RLSPI algoirthm presented in the next section.

4. Robust Least Squares Policy Iteration

In this section, we introduce the robust least squares policy iteration (RLSPI) algorithm for finding the optimal robust policy. RLSPI algorithm can be thought as the robust version of the LSPI algorithm (Lagoudakis & Parr, 2003). RLSPI algorithm uses the RLSPE(λ) algorithm for policy evaluation. However, model-free policy improvement is difficult when working with value functions since the policy update step will require us to solve

$$\pi_{k+1} = \arg\max_\pi \widetilde{T}_\pi^{(\lambda)}(\bar{V}_k), \quad (24)$$

where $\bar{V}_k$ is the approximate robust value function corresponding to the policy π_k, in the $(k+1)^{\text{th}}$ policy iteration loop. To overcome this, we first introduce the robust state-action value function (Q-function).

For any given policy π and state-action pair (s,a), we define the robust Q-value as,

$$Q_\pi(s,a) = \inf_{P\in\mathcal{P}} \mathbb{E}_P[\sum_{t=0}^{\infty}\alpha^t r(s_t,a_t) \,|s_0 = s, a_0 = a]. \quad (25)$$

Instead of learning the approximate robust value function $\bar{V}_\pi$, we can learn the approximate robust Q-value function $\bar{Q}_\pi$ using RLSPE(λ). This can be done by defining the feature vector $\phi(s,a)$ where $\phi(s,a) = (\phi_1(s,a),\ldots,\phi_L(s,a))^\top$ and the linear approximation of the form $\bar{Q}_\pi(s,a) = w^\top\phi(s,a)$ where w is a weight vector. The results from the previous section on the convergence of the RLSPE(λ) algorithm applies for the case of learning Q-value function as well.

RLSPI is a policy iteration algorithm that uses RLSPE(λ) for policy evaluation at each iteration. It starts with an arbitrary initial policy π_0. At the kth iteration, RLSPE(λ) returns a weight vector that represents the approximate Q-value function $\bar{Q}_{\pi_k} = \Phi w_{\pi_k}$ corresponding to the policy π_k. The next policy π_{k+1} is the greedy policy corresponding to $\bar{Q}_{\pi_k}$, defined as $\pi_{k+1}(s) = \arg\max_{a\in\mathcal{A}}\bar{Q}_{\pi_k}(s,a)$. For empirical evaluation purposes, we terminate the policy iteration for some finite value K. RLSPI algorithm is summarized in Algorithm 1.

We make the following assumptions for the convergence analysis of the RLSPI algoirthm. We note that we work with value functions instead of Q-value functions for notational convenience and consistency.

Assumption 3. *(i) Each policy π_k is an exploration policy, i.e. $\pi_{exp}(\pi_k) = \pi_k$.*
(ii) The Markov chain $P^o_{\pi_k}$ has a stationary distribution d_{π_k}

Algorithm 1 RLSPI Algorithm

Initialization: Policy evaluation weights error ϵ_0, initial policy π_0.
for $k = 0 \dots K$ **do**
  Initialize the policy weight vector w_0. Initialize time step $t \leftarrow 0$.
  repeat
    Observe the state s_t, take action $a_t = \pi_k(s_t)$, observe reward r_t and next state s_{t+1}.
    Update the weight vector w_t according to RLSPE(λ) algorithm (c.f. (18)-(23))
    $t \leftarrow t + 1$
  until $\|w_t - w_{t-1}\|_2 < \epsilon_0$
  $w_{\pi_k} \leftarrow w_t$
  Update the policy

$$\pi_{k+1}(s) = \arg\max_{a \in \mathcal{A}} \phi(s,a)^\top w_{\pi_k}$$

end for

such that $d_{\pi_k}(s) > 0, \forall s \in \mathcal{S}$.
(iii) There exists a finite scalar δ *such that* $\|V_{\pi_k} - \Pi_{d_{\pi_k}} V_{\pi_k}\|_{d_{\pi_k}} < \delta$ *for all* k, *where* $\Pi_{d_{\pi_k}}$ *is a projection onto the subspace spanned by the columns of* Φ *under the* d_{π_k}*-weighted Euclidean norm.*
(iv) For any probability distribution μ, *define another probability distribution* $\mu_k = \mu H_k$ *where* H_k *is a stochastic matrix defined with respect to* π_k. *Also assume that there exists a probability distribution* $\bar{\mu}$ *and finite positive scalars* C_1, C_2 *such that* $\mu_k \leq C_1 \bar{\mu}$ *and* $d_{\pi_k} \geq \bar{\mu}/C_2$ *for all* k.

We note that these are the standard assumptions used in the RL literature to provide theoretical guarantees for approximate policy/value iteration algorithms with linear function approximation in the non-robust settings (Munos, 2003; Munos & Szepesvári, 2008; Lazaric et al., 2012). We make no additional assumptions even though we are addressing the more difficult robust RL problem. The specific form of the stochastic matrix H_k specified in Assumption 3.(iv) is deferred to the proof of Theorem 3 for brevity of the presentation.

We now give the asymptotic convergence result for the RLSPI algorithm. We assume that, similar to the non-robust setting (Munos, 2003), the policy evaluation step (inner loop) is run to the convergence. We only present the case where $\rho = 0$. The proof for the general case is straightforward, but involves much more detailed algebra. So, we omit those details for the clarity of presentation.

Theorem 3. *Let Assumption 2 and Assumption 3 hold. Let* $\{\pi_k\}$ *be the sequence of the policies generated by the RLSPI algorithm. Let* V_{π_k} *and* $\bar{V}_k = \Phi w_{\pi_k}$ *be true robust value function and the approximate robust value function corresponding to the policy* π_k. *Also, let* V^* *be the optimal robust value function. Then, with* $c(\alpha, \beta, 0, \lambda) < 1$,

$$\limsup_{k\to\infty} \|V^* - V_{\pi_k}\|_\mu \leq \frac{2\sqrt{C_1 C_2}\, c(\alpha,\beta,0,\lambda)}{(1 - c(\alpha,\beta,0,\lambda))^2} \limsup_{k\to\infty} \|V_{\pi_k} - \bar{V}_k\|_{d_{\pi_k}}. \quad (26)$$

Moreover, from Theorem 1 and Assumption 3.(iii)*, we have*

$$\limsup_{k\to\infty} \|V^* - V_{\pi_k}\|_\mu \leq \frac{2\sqrt{C_1 C_2}\, c(\alpha,\beta,0,\lambda)}{(1 - c(\alpha,\beta,0,\lambda))^3}\, \delta. \quad (27)$$

The above theorem, in particular (27), gives a (worst case) guarantee for the performance of the policy learned using the RLSPI algorithm. Note that the upper bound in (27) is a constant where δ represents the (unavoidable) error due to the linear function approximation. We also note that using 'lim sup' is necessary due to the policy chattering phenomenon in approximate policy iteration algorithms which exists even in the non-robust case (Bertsekas, 2012).

Instead of the asymptotic bound given in Theorem 3, we can actually get a bound for any K given in the RLSPI algorithm by modifying Assumption 3.(iv). We defer the precise statements of the assumption and theorem to Section C in the supplementary material due to page limitation.

5. Experiments

We implemented our RLSPI algorithm using the MushroomRL library (D'Eramo et al., 2020), and evaluated its performance against Q-learning algorithm for an environment with discrete action space, deep deterministic policy gradient (DDPG) (Lillicrap et al., 2016) algorithm for continuous action space, and LSPI algorithm (Lagoudakis & Parr, 2003). For comparing with the performance of our RLSPI algorithm against another robust RL algorithm, we implemented the soft-robust algorithms proposed in (Derman et al., 2018) which use deep neural networks for function approximation.

We chose a spherical uncertainty set with a radius r. For such a set $\widehat{\mathcal{U}}$, a closed form solution of $\sigma_{\widehat{\mathcal{U}}}(\Phi w)$ can be computed for faster simulation. We note that in all the figures shown below, the quantity in the vertical axis is averaged over 100 runs, with the thick line showing the averaged value and the band around shows the ± 0.5 standard deviation. These figures act as the performance criteria for comparing results. We provide more details and additional experiment results in Section D of supplementary.

We used the CartPole, MountainCar, and Acrobot environments from OpenAI Gym (Brockman et al., 2016). We trained LSPI algorithm and our RLSPI algorithm on these environments with nominal parameters (default parameters in OpenAI Gym (Brockman et al., 2016)). We also

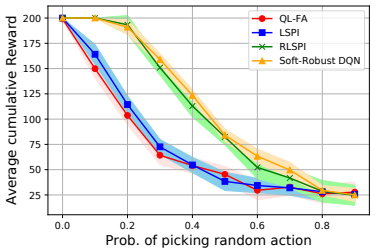

Figure 1: CartPole

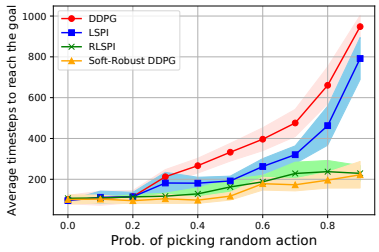

Figure 2: MoutainCar

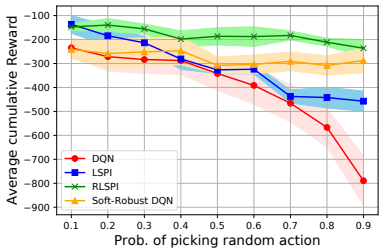

Figure 3: Acrobot

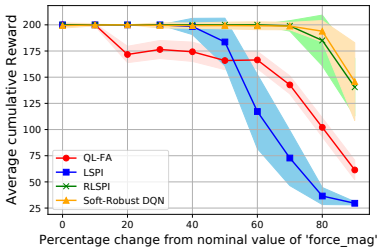

Figure 4: CartPole

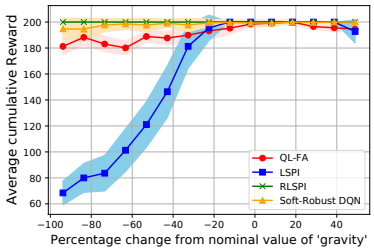

Figure 5: CartPole

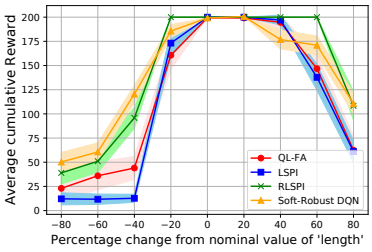

Figure 6: CartPole

trained Q-learning with linear function approximation and soft-robust deep Q-network(DQN) (Derman et al., 2018) algorithms on CartPole environment, DQN and soft-robust DQN (Derman et al., 2018) algorithms on Acrobot environment, and DDPG and soft-robust DDPG (Derman et al., 2018) algorithms on MountainCar environment. Then, to evaluate the robustness of the polices obtained, we changed the parameters of these environments and tested the performance of the learned polices on the perturbed environment.

In Figures 1-3, we show the robustness against action perturbations. In real-world settings, due to model mismatch or noise in the environments, the resulting action can be different from the intended action. We model this by picking a random action with some probability at each time step. Figure 1 shows the change in the average episodic reward against the probability of picking a random action for the CartPole environment. Figure 2 shows the average number of time steps to reach the goal in the MountainCar environment. Figure 3 shows the average episodic reward in the Acrobot environment. In all three cases, RLSPI algorithm shows robust performance against the perturbations.

Figures 4-6 show the test performance on CartPole, by changing the parameters *force_mag* (external force disturbance), *gravity*, *length* (length of pole on the cart). The nominal values of these parameters are $10, 9.8$, and 0.5 respectively. RLSPI again exhibits robust performance.

The performance of our RLSPI algorithm is consistently superior to that of the non-robust algorithms. Moreover, the performance of RLSPI algorithm is comparable with that of the soft-robust algorithms (Derman et al., 2018), even though the latter uses deep neural networks for function approximation while our algorithm uses only linear function approximation architecture. We also would like to emphasize that our work gives provable guarantees for the policy learned by the algorithm whereas (Derman et al., 2018) does not provide any such guarantees.

6. Conclusion and Future Work

We have presented an online model-free reinforcement learning algorithm to learn control policies that are robust to the parameter uncertainties of the model, for system with large state spaces. While there have been interesting empirical works on robust deep RL using neural network, they only provide convergence guarantees to a local optimum. Different from such empirical works, we proposed a learning based robust policy iteration algorithm called RLSPI algorithm with explicit theoretical guarantees on the performance of the learned policy. To the best of our knowledge, this is the first work that presents model-free reinforcement learning algorithm with function approximation for learning the optimal robust policy. We also empirically evaluated the performance of our RLSPI algorithm on standard benchmark RL problems.

In future, we plan to extend our theoretical results to nonlinear function approximation architectures. We also plan to characterize the sample complexity of robust reinforcement learning algorithms. Extending offline RL approaches to robust setting is another research area that we plan to pursue.

Acknowledgments

Dileep Kalathil gratefully acknowledges funding from the U.S. National Science Foundation (NSF) grants NSF-CRII-CPS-1850206 and NSF-CAREER-EPCN-2045783.

References

Abbeel, P. and Ng, A. Y. Apprenticeship learning via inverse reinforcement learning. In *International Conference on Machine Learning*, pp. 1, 2004.

Antos, A., Szepesvári, C., and Munos, R. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008.

Arora, S., Du, S., Kakade, S., Luo, Y., and Saunshi, N. Provable representation learning for imitation learning via bi-level optimization. In *International Conference on Machine Learning*, pp. 367–376. PMLR, 2020.

Bertsekas, D. P. Approximate policy iteration: A survey and some new methods. *Journal of Control Theory and Applications*, 9(3):310–335, 2011.

Bertsekas, D. P. *Dynamic programming and optimal control, Vol - 2*. Athena scientific Belmont, MA, 2012.

Bertsekas, D. P. and Ioffe, S. Temporal Differences-Based Policy Iteration and Applications in Neuro-Dynamic Programming. *Lab. for Info. and Decision Systems Report LIDS-P-2349, MIT, Cambridge, MA*, 14, 1996.

Bertsekas, D. P. and Yu, H. Projected equation methods for approximate solution of large linear systems. *Journal of Computational and Applied Mathematics*, 227(1):27–50, 2009.

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

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

D'Eramo, C., Tateo, D., Bonarini, A., Restelli, M., and Peters, J. Mushroomrl: Simplifying reinforcement learning research. *arXiv preprint arXiv:2001.01102*, 2020. URL https://github.com/MushroomRL/mushroom-rl.

Derman, E., Mankowitz, D. J., Mann, T. A., and Mannor, S. Soft-robust actor-critic policy-gradient. In *AUAI press for Association for Uncertainty in Artificial Intelligence*, pp. 208–218, 2018.

Duan, Y., Jia, Z., and Wang, M. Minimax-optimal off-policy evaluation with linear function approximation. In *International Conference on Machine Learning*, pp. 2701–2709. PMLR, 2020.

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

Jin, C., Yang, Z., Wang, Z., and Jordan, M. I. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pp. 2137–2143, 2020.

Kaufman, D. L. and Schaefer, A. J. Robust modified policy iteration. *INFORMS Journal on Computing*, 25(3):396–410, 2013.

Kong, W., Somani, R., Song, Z., Kakade, S., and Oh, S. Meta-learning for mixed linear regression. In *International Conference on Machine Learning*, pp. 5394–5404. PMLR, 2020.

Lagoudakis, M. G. and Parr, R. Least-squares policy iteration. *Journal of Machine Learning Research*, 4(Dec): 1107–1149, 2003.

Lazaric, A., Ghavamzadeh, M., and Munos, R. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13(Oct):3041–3074, 2012.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In *ICLR (Poster)*, 2016.

Lim, S. H. and Autef, A. Kernel-based reinforcement learning in robust Markov decision processes. In *International Conference on Machine Learning*, pp. 3973–3981, 2019.

Lim, S. H., Xu, H., and Mannor, S. Reinforcement learning in robust Markov decision processes. In *Advances in Neural Information Processing Systems*, pp. 701–709, 2013.

Mankowitz, D. J., Levine, N., Jeong, R., Abdolmaleki, A., Springenberg, J. T., Shi, Y., Kay, J., Hester, T., Mann, T., and Riedmiller, M. Robust reinforcement learning for continuous control with model misspecification. In *International Conference on Learning Representations*, 2020.

Munos, R. Error bounds for approximate policy iteration. In *ICML*, volume 3, pp. 560–567, 2003.

Munos, R. and Szepesvári, C. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9 (27):815–857, 2008.

Nedić, A. and Bertsekas, D. P. Least squares policy evaluation algorithms with linear function approximation. *Discrete Event Dynamic Systems*, 13(1-2):79–110, 2003.

Nilim, A. and El Ghaoui, L. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

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

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., NJ, 2005.

Roy, A., Xu, H., and Pokutta, S. Reinforcement learning under model mismatch. In *Advances in Neural Information Processing Systems*, pp. 3043–3052, 2017.

Tamar, A., Mannor, S., and Xu, H. Scaling up robust mdps using function approximation. In *International Conference on Machine Learning*, pp. 181–189, 2014.

Tessler, C., Efroni, Y., and Mannor, S. Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning*, pp. 6215–6224, 2019.

Tsitsiklis, J. N. and Van Roy, B. Analysis of temporal-difference learning with function approximation. In *Advances in Neural Information Processing Systems*, pp. 1075–1081, 1997.

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

Wang, L., Cai, Q., Yang, Z., and Wang, Z. On the global optimality of model-agnostic meta-learning. In *International Conference on Machine Learning*, pp. 9837–9846. PMLR, 2020.

Wang, R., Foster, D., and Kakade, S. M. What are the statistical limits of offline {rl} with linear function approximation? In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=30EvkP2aQLD.

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

Xu, H. and Mannor, S. Distributionally robust Markov decision processes. In *Advances in Neural Information Processing Systems*, pp. 2505–2513, 2010.

Yang, L. and Wang, M. Reinforcement learning in feature space: Matrix bandit, kernels, and regret bound. In *International Conference on Machine Learning*, pp. 10746–10756. PMLR, 2020.

Yu, P. and Xu, H. Distributionally robust counterpart in Markov decision processes. *IEEE Transactions on Automatic Control*, 61(9):2538–2543, 2015.

Zhang, H., Chen, H., Xiao, C., Li, B., Boning, D., and Hsieh, C.-J. Robust deep reinforcement learning against adversarial perturbations on observations. In *Advances in Neural Information Processing Systems*, 2020.